Appendix A – Claim Chart for US Patent No. 9,027,032 Against Accused Cisco Products

Based on information presently available, Damaka Inc. ("Damaka") contends that Cisco Systems, Inc. ("Cisco" or "Defendant") infringes Claims 15-20 (the "Asserted Claims") of U.S. Patent No. 9,072,032 (the "'032 Patent") through the Accused Products which are manufactured, sold, offered for sale, used, imported into the United States, and/or exported from the United States by Cisco.

The Accused Products comprise all versions and variants of the Webex Application and Webex SDKs[1], including, but not limited to, Webex for iOS and Webex for Android that are made, used, sold, offered for sale, imported into the United States, and/or exported from the United States or otherwise provided, after May 30, 2019. Damaka reserves the right to amend this list of Accused Products as discovery progresses.

Cisco directly infringes each of the Asserted Claims by using, importing, testing, selling, offering for sale, importing into the United States, and/or exporting from the United States the Accused Products in violation of 35 U.S.C. § 271(a).

Cisco indirectly infringes the Asserted Claims in violation of 35 U.S.C. § 271(b) by inducing third parties, including Cisco's customers and end-users, to directly infringe through their operation and use of the Accused Products. Cisco has knowingly and intentionally induced this direct infringement by, *inter alia*, manufacturing, selling, distributing, and/or otherwise making available the Accused Products, and providing instructions, documentation, and other information to customers and end-users suggesting that they use the Accused Products in an infringing manner, including technical support, marketing, product manuals, advertisements, and online documentation. Cisco's marketing and promotional materials for the Accused Products are found, for example, on the Webex website (https://www.webex.com/). For example, the Webex website instructs customers to, among other things, use the accused in the Accused Products. *See* https://www.Webex.com/frequently-asked-questions. The Webex website also offers support to customers, including instruction on, among other things, how to operate their products. On information and belief, Cisco knows that its actions will result in infringement of the Asserted Claims, or subjectively believes that there is a high probability that its actions will result in infringement of the Asserted Claims but has taken deliberate actions to avoid learning these facts.

Cisco also contributorily infringes each of the Asserted Claims in violation of 35 U.S.C. § 271(c) by selling, importing, offering for sale, and otherwise providing the Accused Products, which when used by third parties, such as customers and end-users, directly infringe the Asserted Claims. The Accused Products constitute a material part of the Asserted Claims, are not staple articles or commodities of commerce, have no substantial non-infringing uses, and are known by Cisco to be especially made or adapted for use in the infringement of the '032 Patent. On information and belief, Cisco knows that its actions will result in infringement of the Asserted Claims, or subjectively believes that there is a high probability that its actions will result in infringement of the Asserted Claims but has taken deliberate actions to avoid learning these facts.

On information and belief, the charted version of the Webex application and Webex SDK is representative of all the Accused Products as functional equivalents of all claims exist in iOS, and Android.[2]

Damaka does not concede that any claims of the '032 Patent that are not listed below are not infringed by the identified Accused Products. Moreover, the citations to certain documents and other information below are intended to be exemplary only and in no way foreclose Damaka from citing or relying on additional documents, information, source code, and/or testimony at a later time. These contentions are preliminary in nature, and an analysis of Cisco's products, internal documentation, source code, and/or testimony from relevant witnesses may more fully and accurately describe the infringing features of its Accused Products. Accordingly, Damaka reserves the right to supplement, correct, modify, and/or amend these contentions once such additional information is made available to Damaka. Furthermore, Damaka reserves the right to supplement, correct, modify, and/or amend

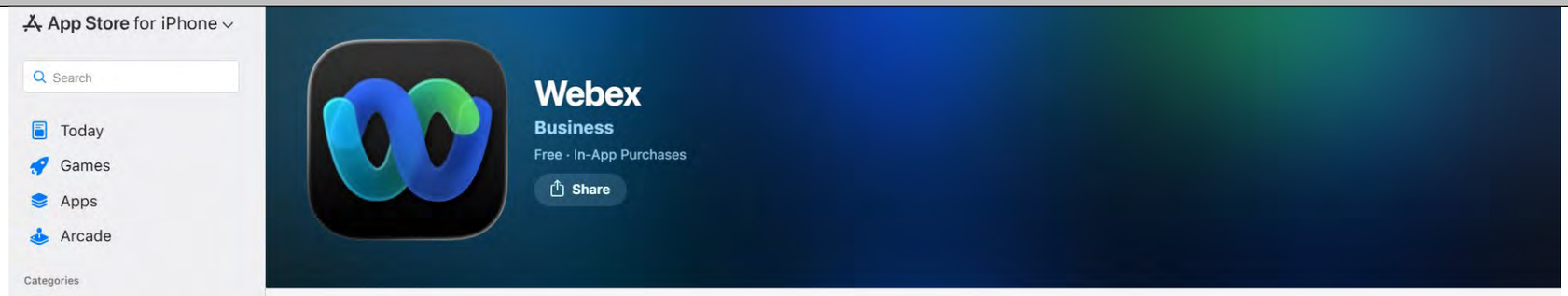
[1] The Webex SDK encompasses both the SDK as and associated Webex API. *See e.g.* https://developer.webex.com/meeting/docs/meetings

[2] *See e.g.* https://developer.webex.com/meeting/docs/meetings; *See also* https://help.webex.com/en-us/article/fz1e4b/System-requirements-for-Webex-services

these contentions as discovery in this case progresses; in view of the Court's claim construction order(s); in view of any positions taken by Cisco, including, but not limited to, positions on claim construction, invalidity, and/or non-infringement; and in connection with the preparation and exchange of expert reports.

To the extent Cisco contends that any element of the Accused Products is attributable to a third party, Damaka contends that the activities are attributable to Cisco such that they constitute direct infringement by Cisco under 271(a). The acts may be attributable to Cisco because Cisco directs or controls the others' performance, and because Cisco and the other entity form a joint enterprise. *Akamai Techs., Inc. v. Limelight Networks, Inc.*, 797 F.3d 1020, 1022 (Fed. Cir. 2015). Additionally, Cisco is vicariously liable for the activities of these other entities. *Centillion Data Sys., LLC v. Qwest Commc'ns Int'l, Inc.*, 631 F.3d 1279, 1286 (Fed. Cir. 2011). Further, the activities of these entities (including third party application developers, and users of the Accused Products consumers), are attributable to Cisco because Cisco (1) conditions participation in an activity or receipt of a benefit upon others' performance of one or more steps of a patented method, and (2) establishes the manner or timing of that performance. *Eli Lilly & Co. v. Teva Parenteral Medicines, Inc.*, 845 F.3d 1357, 1365 (Fed. Cir. 2017). These acts are also attributed to Cisco because it initiated the activities of their end-users. *SiRF Tech., Inc. v. Int'l Trade Comm'n*, 601 F.3d 1319, 1330 (Fed. Cir. 2010).

Claim	Cisco Accused Products
15.p A method comprising:	Each Accused Product practices a method, said method comprising the elements of claim 1. Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
15a. receiving, by a function block stored in a memory of a mobile device, a request for an audio/video (A/V) call from a superblock application stored in the memory of the mobile device;	Each Accused Product literally performs the step of "receiving, by a function block stored in a memory of a mobile device, a request for an audio/video (A/V) call from a superblock application stored in the memory of the mobile device." For example, Cisco's Webex (superblock application) is a mobile app stored in the memory of a mobile device.

Claim	Cisco Accused Products
	 https://apps.apple.com/us/app/webex/id833967564 For example, the Webex SDK (function block) receives a call from Webex (the superblock application) for A/V capabilities. As shown below "Webex Meetings [SDK] offers integrated audio, video…" **Webex Meetings** Webex Meetings offers integrated audio, video, and content sharing with highly secure web meetings from the cloud. The Webex Meetings REST API allows developers to add basic Webex scheduling functionality to their custom applications or websites. You can: • Create and manage meetings • Add and manage meeting invitees • Get and update meeting preferences, options, and other details • Get, list, and delete recordings https://developer.webex.com/meeting/docs/meetings

Claim	Cisco Accused Products
	For example, the function block (Webex SDK) can receive a “Join_a_meeting” call. This allows the superblock application “to start or join a meeting” via the function block.

	# Join a Meeting **Operation Id:** *Join_a_Meeting* **Description:** *Retrieves links for a meeting with a specified `meetingId`, `meetingNumber`, or `webLink` that allow users to start or join the meeting directly without logging in and entering a password.* • *Please note that `meetingId`, `meetingNumber` and `webLink` are mutually exclusive and they cannot be specified simultaneously.* • *If `joinDirectly` is true or not specified, the response will have HTTP response code 302 and the request will be redirected to `joinLink`; otherwise, the response will have HTTP response code 200 and `joinLink` will be returned in response body.* • *Only the meeting host or cohost can generate the `startLink`.* • *An admin user or a Service App can generate the `startLink` and `joinLink` on behalf of another meeting host using the `hostEmail` parameter. When a Service App generates the `startLink` and `joinLink`, the `hostEmail` parameter is required. The `hostEmail` parameter only applies to meetings, not webinars.* • *For Service Apps, `hostEmail` must be provided in the request.* • *Generating a join link or a start link before the time specified by `joinBeforeHostMinutes` for a webinar is not supported.* • *The `joinLink` and `startLink` generated by the API only work in a web browser, not in mobile apps. When the `joinLink` or `startLink` is used in a mobile app, the user can't join or start the meeting directly by the dial-in or dial-out phone numbers displayed in the app.* • *When `createJoinLinkAsWebLink` or `createStartLinkAsWebLink` is set to true, a user cannot join or start the meeting using the `joinLink` or `startLink` returned in the response, and must complete the login flow. Those options are typically useful if mandatory user login is configured in Control Hub.* • *Appending the parameter `&launchApp=true` to the weblink will launch directly into the browser experience without showing the user the download selection for the native client.* • *Embedded Apps are not supported in a meeting which is started from the `startLink` generated by this API.*

Claim	Cisco Accused Products

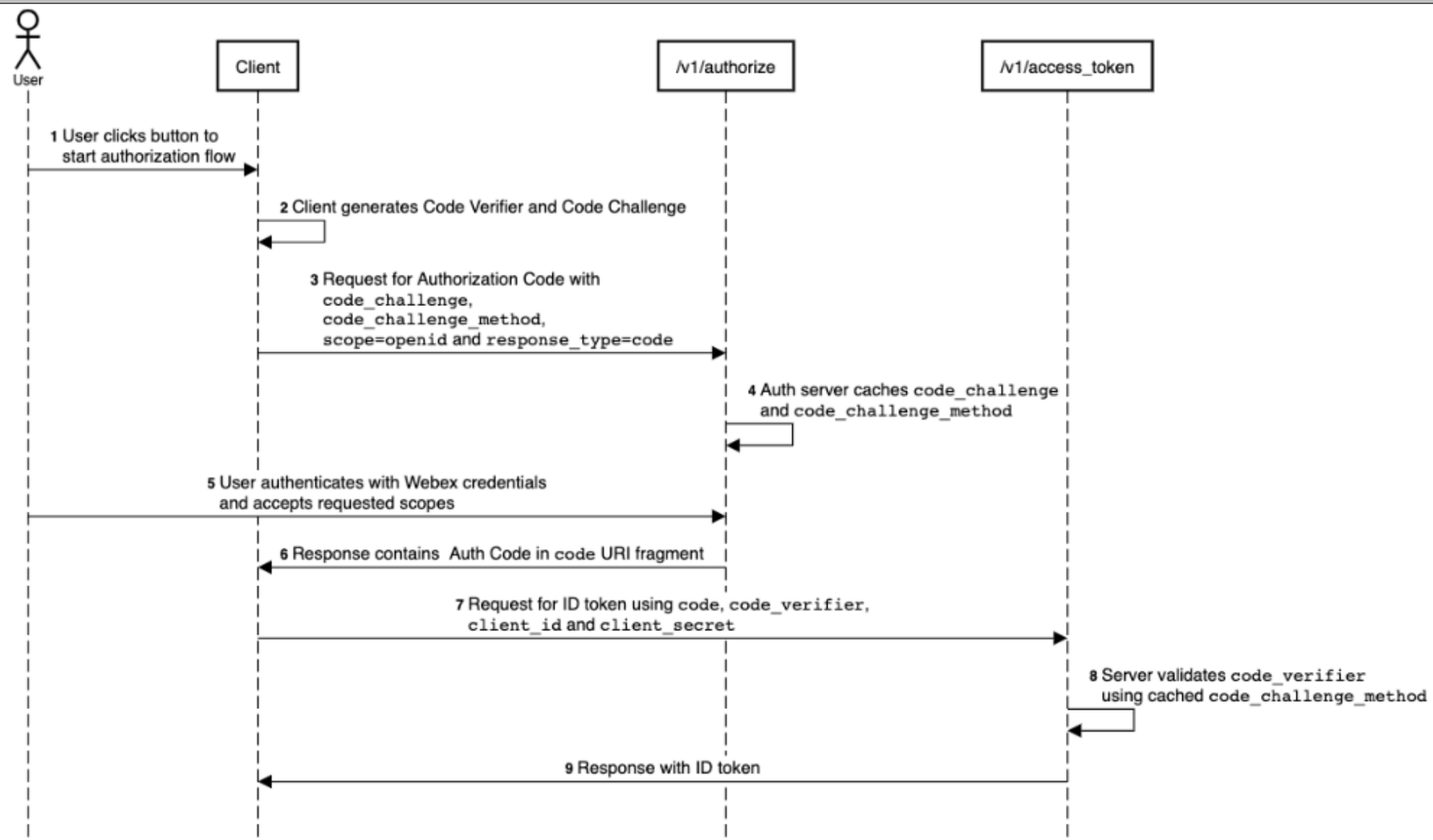
	https://developer.webex.com/meeting/docs/api/v1/meetings/join-a-meeting
	
	There is also equivalent functionality in the formal SDK, as demonstrated by the below.
	
	Fetch a Meeting Instance
	
	A `meeting` instance is typically created once a meeting is initiated through the SDK. If a meeting has already been created on the SDK, you can retrieve it using the SDK's `meetings` object by passing a `meetingId`.
	
	This method does not fetch the list of all meetings from the Webex cloud. It only lists the meetings that are created either with `webex.meetings.create` or `webex.meetings.syncMeetings`.
	
	``
	const meeting = webex.meetings.getAllMeetings()[meetingId];
	``
	
	If participants have already joined the meeting, it can be retrieved from Webex Cloud. Refer to Sync Meetings.
	
	Join a Meeting
	
	Once you have a `Meeting` object, you can join the meeting using the `join()` method:
	
	``
	const join = await meeting.join(joinOptions);
	``
	
	https://developer.webex.com/meeting/docs/sdks/webex-meetings-sdk-web-join-a-meeting

Claim	Cisco Accused Products
	Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
15b. determining, by the function block, that the superblock application is authorized to access A/V call functionality provided by the function block;	Each Accused Product literally performs the step of "determining, by the function block, that the superblock application is authorized to access A/V call functionality provided by the function block." For example, to successfully connect to a meeting, the Webex SDK checks authorization of superblock application (*e.g.* Webex), and if not authorized, the SDK will throw a 401 error, as shown below **Status:** 401 *Unauthorized: Authentication credentials were missing or incorrect.* https://developer.webex.com/meeting/docs/api/v1/meetings/join-a-meeting Again, there is analogous functionality within the formal Webex SDK.

Claim	Cisco Accused Products

	Initialize the SDK
	
	To use the SDK, you'll need a free Webex account; if you don't have a Cisco Webex account, visit Cisco Webex for Developers to create your account and retrieve your **access token** from Getting and Using your Personal Access Token.
	
	To create a new Webex instance:
	
	`const webex = Webex.init({`
	`  credentials: {`
	`  access_token: `<AUTHORIZATION/BOT TOKEN>``
	`  }`
	`});`
	
	At this point, the Webex `Meetings` object is ready for use. The `Meetings` object contains a list of meetings as well as methods to perform specific operations or listen for certain events. Detailed information is included in this article.
	
	https://developer.webex.com/meeting/docs/sdks/webex-meetings-sdk-authorization-device-registration
	
	For example, authentication credentials must be acquired to use the Webex SDK. Without proper authorization, Webex SDK will not function, as evidenced by the below.

Claim	Cisco Accused Products
	What Are Integrations? Integrations are how you request permission to invoke the Webex REST API on behalf of another Webex user. To do this in a secure way the API supports the OAuth 2 standard which allows third-party integrations to get a temporary access token for authenticating API calls instead of asking users for their password. If you're sure that your integrations require authenticating on behalf of another Webex user, read on, we'll get you there in a few easy steps: 1. Register your integration with Webex 2. Request permission using an OAuth Grant Flow 3. Exchange the resulting authorization code for an access token 4. Use the access token to make your API calls https://developer.webex.com/create/docs/integrations Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
15c. determining, by the function block, that the A/V call requires access to a resource that is external to the mobile device;	Each Accused Product literally performs the step of "determining, by the function block, that the A/V call requires access to a resource that is external to the mobile device." For example, when authorizing the superblock application, the function block (Webex SDK) must contact Cisco servers that are external to the mobile device the Webex application is on. As shown below, the function block (Webex SDK) makes a request to a Cisco server to "validate[]."

Claim	Cisco Accused Products
	

Claim	Cisco Accused Products
	https://developer.webex.com/create/docs/login-with-webex If authorization is not successful, the function block will throw a 401 error, as shown in claim 15b. *See* support for claim 15b. Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
15d. establishing, by the function block, the A/V call between the superblock application and the resource via a network interface, wherein the establishing is not supported by the superblock application;	Each Accused Product literally performs the step of "establishing, by the function block, the A/V call between the superblock application and the resource via a network interface, wherein the establishing is not supported by the superblock application." For example, the function block needs to send the A/V request over the internet using the network interface of the mobile device. *See* claim 15a and 15b for support. Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
15e. providing, by the function block, video output for the A/V call to a video window within a display window of the superblock application on the mobile device,	Each Accused Product literally performs the step of "providing, by the function block, video output for the A/V call to a video window within a display window of the superblock application on the mobile device, wherein the video window is displayed simultaneously with the display window of the superblock application on a screen of the mobile device."

Claim	Cisco Accused Products
wherein the video window is displayed simultaneously with the display window of the superblock application on a screen of the mobile device; and	For example, the Webex SDK provides a video window within the Webex Application, as shown below.

Claim	Cisco Accused Products

Claim	Cisco Accused Products
	https://www.webex.com/free-video-conferencing The Webex SDK includes functionality for display within the superblock application, for example "Get_Video_Options…Retrieves video options." Preferences Get Video Options Operation Id: *Get_Video_Options* Description: *Retrieves video options for the authenticated user* GET /meetingPreferences/video Request Parameters https://developer.webex.com/meeting/docs/api/v1/meeting-preferences/get-video-options These video options can then be updated by the function block via the "Update_video_options" call.

Claim	Cisco Accused Products

	Update Video Options
	Operation Id: *Update_Video_Options*
	Description: *Updates video options for the authenticated user.*
	PUT /meetingPreferences/video
	Request Parameters
	https://developer.webex.com/meeting/docs/api/v1/meeting-preferences/update-video-options
	
	Again, there is analogous functionality within the formal Webex SDK.
	
	Pass one of the above-desired profile levels to the `Meeting` object's `setRemoteQualityLevel()` method to set the remote video resolution:
	`meeting.setRemoteQualityLevel('HIGH');`
	Or
	`meeting.setRemoteQualityLevel('LOW');`
	https://developer.webex.com/meeting/docs/sdks/webex-meetings-sdk-web-video-resolution

Claim	Cisco Accused Products

	Streams
	
	The following new `Streams` classes have been introduced:
	
	1. LocalCameraStream
	2. LocalMicrophone
	3. LocalDisplayStream
	4. LocalSystemAudioStream
	
	Import the `Streams` classes into your applications using the `plugin-meetings` package:
	
	``
	import {
	  LocalCameraStream,
	  LocalMicrophoneStream,
	  LocalDisplayStream,
	  LocalSystemAudioStream,
	  createCameraStream,
	  createMicrophoneStream,
	  createDisplayStream,
	  createDisplayStreamWithAudio
	} from '@webex/plugin-meetings';
	``
	
	https://developer.webex.com/meeting/docs/sdks/webex-meetings-sdk-web-web-streams-and-effects
	
	Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.

Claim	Cisco Accused Products
15f. providing, by the function block, audio for the A/V call via the mobile device.	Each Accused Product literally performs the step of "providing, by the function block, audio for the A/V call via the mobile device." As stated in claim 15[b], "Webex Meetings [SDK] offers integrated audio, video…" **Webex Meetings** Webex Meetings offers integrated audio, video, and content sharing with highly secure web meetings from the cloud. The Webex Meetings REST API allows developers to add basic Webex scheduling functionality to their custom applications or websites. You can: • Create and manage meetings • Add and manage meeting invitees • Get and update meeting preferences, options, and other details • Get, list, and delete recordings https://developer.webex.com/meeting/docs/meetings For example, the Webex SDK provides "Get_Audio_Options" and "Update_Audio_Options" functionality to the superblock application.

Claim	Cisco Accused Products
	Preferences **Get Audio Options** **Operation Id:** *Get_Audio_Options* **Description:** *Retrieves audio options for the authenticated user.* GET /meetingPreferences/audio Request Parameters https://developer.webex.com/meeting/docs/api/v1/meeting-preferences/get-audio-options Preferences **Update Audio Options** **Operation Id:** *Update_Audio_Options* **Description:** *Updates audio options for the authenticated user.* PUT /meetingPreferences/audio Request Parameters https://developer.webex.com/meeting/docs/api/v1/meeting-preferences/update-audio-options

Claim	Cisco Accused Products

	For example, the formal SDK also provides access to "LocalMicroPhoneStream" to provide the same functionality.
	
	Streams
	
	The following new `Streams` classes have been introduced:
	
	1. LocalCameraStream
	2. LocalMicrophone
	3. LocalDisplayStream
	4. LocalSystemAudioStream
	
	Import the `Streams` classes into your applications using the `plugin-meetings` package:
	
	``
	import {
	  LocalCameraStream,
	  LocalMicrophoneStream,
	  LocalDisplayStream,
	  LocalSystemAudioStream,
	  createCameraStream,
	  createMicrophoneStream,
	  createDisplayStream,
	  createDisplayStreamWithAudio
	} from '@webex/plugin-meetings';
	``
	
	https://developer.webex.com/meeting/docs/sdks/webex-meetings-sdk-web-web-streams-and-effects

Claim	Cisco Accused Products
	Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
16. The method of claim 15 further comprising determining, by the function block, whether the function block has the capability to provide the A/V call.	Each Accused Product literally performs the step of "determining, by the function block, whether the function block has the capability to provide the A/V call." For example, if the Webex SDK cannot provide A/V call capabilities, for example the join meeting function will throw a 401 error if it is unauthorized, and a 403 error if the function block is forbidden.

Claim	Cisco Accused Products
	Status: 400 *Bad Request: The request was invalid or cannot be otherwise served. An accompanying error message will explain further.* **Status:** 401 *Unauthorized: Authentication credentials were missing or incorrect.* **Status:** 403 *Forbidden: The request is understood, but it has been refused or access is not allowed.* **Status:** 404 *Not Found: The URI requested is invalid or the resource requested, such as a user, does not exist. Also returned when the requested format is not supported by the requested method.* **Status:** 405 *Method Not Allowed: The request was made to a resource using an HTTP request method that is not supported.* https://developer.webex.com/meeting/docs/api/v1/meetings/join-a-meeting *See* support for claim 15.

Claim	Cisco Accused Products
	Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
17. The method of claim 16 further comprising determining, by the function block, whether the function block has the capability to render the A/V call.	Each Accused Product literally performs the step of "determining, by the function block, whether the function block has the capability to render the A/V call." For example, the Webex SDK has the following resolution capabilities.

Claim	Cisco Accused Products
	Prerequisites Keep the following prerequisites in mind: • The device camera must support **full HD (1080p)**; if not, the feature is downgraded to the camera's maximum resolution support. To check for FHD support, see Webcam Test Resolution. • Make sure that you're using version webex-js-sdk@2.32.0 or above. For SDK versions 2.32.0 and above, the value mappings used for resolutions are LOW: 480p, MEDIUM: 720p, and HIGH 1080p. For earlier SDK versions: LOW: 360p, MEDIUM: 480p, and HIGH: 720p. Local Resolution The sharpness, smoothness, and overall quality of the video are directly linked to the frame rate, bitrate, and other video settings. The following table specifies the different resolution settings available. **Supported Sending Video Resolutions** Video Profile: 360p; Resolution (Width×Height): 640 × 360; Frame Rate (fps): 30; Chrome: ✓; Firefox: ✓; Safari: ✓ 480p; 640 × 480; 30; ✓; ✓; ✓ 720p; 1280 × 720; 30; ✓; ✓; ✓ 1080p; 1920 × 1080; 30; ✓; ✓; ✓

Claim	Cisco Accused Products
	https://developer.webex.com/meeting/docs/sdks/webex-meetings-sdk-web-video-resolution For example, the Webex SDK will throw the following errors if either the superblock or function block cannot render the A/V call: **Remote Video Resolution Errors** The following table lists possible errors when working with remote video resolutions: `Method` / `Error` / `Fix` `setLocalVideoTrack()` — `Meeting:index#setLocalVideoTrack --> Local video quality of "${localQualityLevel}" not supported, downscaling to the highest possible resolution of "${height}p"` — Remote resolution is set to 720p. `setRemoteVideoConstraints()` — `setRemoteVideoConstraints: unable to set max framesize, value for level "${level}" is not defined` — Use the defined resolution values. `getMediaStreams()` — `ParameterError: ${audioVideo} not supported. Either pass level from pre-defined resolutions or pass complete audioVideo object` — Pass a level from the pre-defined resolutions or use a complete `audioVideo` object with appropriate video properties `getUserMedia()` — `Media:index#getMedia --> navigator.mediaDevices.getUserMedia failed - [object OverconstrainedError] (facingMode)` — If you request the rear camera on a laptop device, you'll get this error. Use either `ideal` instead of `exact` or request only cameras appropriate to the device. https://developer.webex.com/meeting/docs/sdks/webex-meetings-sdk-web-video-resolution *See* support for claim 16.

Claim	Cisco Accused Products
	Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
18. The method of claim 17 further comprising determining, by the function block, whether the superblock has the capability to render the A/V call.	Each Accused Product literally performs the step of "determining, by the function block, whether the superblock has the capability to render the A/V call." For example, if either the superblock or function block cannot render the A/V call **Remote Video Resolution Errors** The following table lists possible errors when working with remote video resolutions: Method: `setLocalVideoTrack()` Error: `Meeting:index#setLocalVideoTrack --> Local video quality of "${localQualityLevel}" not supported, downscaling to the highest possible resolution of "${height}p"` Fix: Remote resolution is set to 720p. Method: `setRemoteVideoConstraints()` Error: `setRemoteVideoConstraints: unable to set max framesize, value for level "${level}" is not defined` Fix: Use the defined resolution values. Method: `getMediaStreams()` Error: `ParameterError: ${audioVideo} not supported. Either pass level from pre-defined resolutions or pass complete audioVideo object` Fix: Pass a level from the pre-defined resolutions or use a complete `audioVideo` object with appropriate video properties Method: `getUserMedia()` Error: `Media:index#getMedia --> navigator.mediaDevices.getUserMedia failed - [object OverconstrainedError] (facingMode)` Fix: If you request the rear camera on a laptop device, you'll get this error. Use either `ideal` instead of `exact` or request only cameras appropriate to the device.

Claim	Cisco Accused Products
	https://developer.webex.com/meeting/docs/sdks/webex-meetings-sdk-web-video-resolution *See* support for claim 17. Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
19. The method of claim 15 further comprising initializing the function block, by the superblock application, when the superblock application is loaded into the memory and executed.	Each Accused Product literally performs the step of "initializing the function block, by the superblock application, when the superblock application is loaded into the memory and executed." For example, on information and belief, the function block is initialized when the Superblock application is loaded into memory and executed. *See* support for claim 15. Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
20. The method of claim 15 wherein the request is an application programming interface (API) call from the superblock application to the function block.	Each Accused Product literally meets the requirement "wherein the request is an application programming interface (API) call from the superblock application to the function block." For example, the Webex SDK includes numerous API functionality as shown throughout claim 15. *See* support for claim 15. Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.