EXHIBIT E

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

In these Infringement Contentions, AGIS Software Development LLC (“AGIS”) contends that at least the following claims of U.S. Patent No. 9,467,838 (the “’838 Patent”) identified below are infringed by the Accused Products (*e.g.*, phones and tablets) which are manufactured, sold, offered for sale, and/or used by Defendants ZTE (TX) Inc., ZTE (USA) Inc., and ZTE Corporation (collectively, “ZTE”).

The Accused Products comprise ZTE products running the Android mobile operating system and manufactured, used, or sold during and after 2011. For example, the Accused Products comprise the following Android-based phones and tablets: Adamant, Agent, Anthem 4G, Aspect, Avail / Merit, Avid 4, Avid 4G, Avid Plus, Avid Plus / Avid 828, Avid Trio / ZFive 2, Awe / Emblem, Axon, Axon 7, Axon 7 Max, Axon 7 mini, Axon Elite, Axon Lux, Axon M, Axon Max, Axon Mini, Axon Pro / Axon, Blade A1, Blade A2, Blade A2 Plus, Blade A2S, Blade A3, Blade A910, Blade C, Blade D Lux, Blade D2, Blade D6, Blade E, Blade Force, Blade G2, Blade L, Blade L3, Blade L5 Plus, Blade Max 3, Blade Max 3 / Max Blue, Blade Q, Blade Q Maxi, Blade Q Mini, Blade Qlux 4G, Blade S6, Blade S6 Lux, Blade S6 Plus, Blade S7, Blade Spark, Blade V580, Blade V6, Blade V7, Blade V7 Lite, Blade V7 Max, Blade V7 Plus, Blade V8, Blade V8 Lite, Blade V8 Mini, Blade V8 Pro, Blade Vantage, Blade Vec 3G, Blade Vec 4G, Blade X, Blade X Max, Blade X3, Blade X5, Blade X9, Blade Z Max, C78, C79, C88, CAPTR II / A210, Chorus, Citrine, Compel, Concord / Midnight, Concord II, CYMBAL LTE, Cymbal LTE (Verizon), Cymbal Z-320, Cymbal Z-320 / Cymbal LTE, Cymbal-C LTE, Cymbal-G LTE, CYMBAL-T, Cymbal-T LTE, Engage, Engage LT / Engage MT, Essenze C70, F160, Fanfare, Fanfare 2, Fanfare 3, Flash, Force, Fury / Director, Geek, Grand Memo II LTE, Grand S Flex, Grand S II, Grand S Pro, Grand S3, Grand X, Grand X 3, Grand X 4, Grand X 4, Grand X Max 2 / Imperial MAX, Grand X Max+, Grand X Quad Lite, Grand XMax, Groove, Hawkeye, Imperial, Imperial II, Jasper LTE, Kis 3 Max, Kis Flex, Majesty, Majesty Pro, Majesty Pro LTE, Majesty Pro Plus LTE, Maven 2, Maven 2 / Sonata 3, Maven 3 / Overture 3, Max, MAX Blue LTE, MAX XL, Max+, Memo, Midnight PRO LTE, MSGM8 II, N919D, Nubia M2, Nubia M2 Lite, Nubia M2 Play, Nubia My Prague, Nubia N1, Nubia N1 lite, Nubia N2, Nubia Prague S, Nubia X6, Nubia Z11, Nubia Z11 Max, Nubia Z11 mini, Nubia Z11 mini S, Nubia Z17, Nubia Z17 Lite, Nubia Z17 mini, Nubia Z17 miniS, Nubia Z17S, Nubia Z7, Nubia Z7 Max, Nubia Z7 mini, Nubia Z9 Classic, Nubia Z9 Elite, Nubia Z9 Exclusive, Nubia Z9 Max, Nubia Z9 mini, Obsidian, Open, Open C, Open II, Overture, Overture 2 / Maven, Prelude / Avail 2, Prelude+, Prestige, Prestige 2, Q519T, Quartz, R225, Reef, Render, Salute, Savvy, Score M / Score, Small Fresh 4, Small Fresh 5, Solar, Sonata / Radiant, Sonata 2 / Paragon, Sonata 3, Source, Speed, Star 1, Star 2, Tempo, Tempo X, TXTM8 3G, V3 Energy Edition, V3 Extreme Edition, V3 Youth Edition, V5, V870, Valet, Vital / Supreme, Warp, Warp 4G, Warp 7, Warp Elite, Warp Sequent, Warp Sync, Whirl, Z221, Z222 / Z223, Z331, Z431 / Altair, Z432 / Altair 2, Z667 / Zinger / Prelude 2 / Flame / Whirl 2, Z998 / Unico LTE, ZFive 2 LTE, ZFive L LTE, Zinger, ZMax, Zmax 2, ZMAX 2 (Unlocked), ZMAX Champ LTE, ZMAX Grand / Champ / Avid 916, ZMAX GRAND LTE, ZMax Pro, and any variants thereof. AGIS reserves the right to amend this list of Accused Products as discovery progresses. For example, AGIS reviewed Android-based products from Android-based handset manufacturers, including three ZTE phones (serial numbers 329F75623FA5; 329F7562388B; and 329F75624913) which are available for inspection at ZTE's request. For example, the Accused Products comprise ZTE products, including but not limited to the phones and tablets as described herein, running the following versions (and all intervening updates and sub-versions) of the Android mobile operating system: Android 2.3, 4.0, 4.1, 4.2, 4.3, 4.4, 5.0, 5.1, 6.0, 7.0, 7.1, 8.0, and 8.1. For example, the Accused Products comprise ZTE products, including but not limited to the phones and tablets as described herein, running any versions of the following Android-based applications and/or software: Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome. For example, the Accused Products comprise ZTE products, including but not limited to the phones and tablets described herein, participating in any networks and/or services related to the execution and/or use of the Android mobile operating system versions and Android-based applications and/or software described herein.

AGIS does not concede that any claims of the '838 Patent that are not listed below are not infringed by the identified products. Moreover, the citations to certain documents and other information below are intended to be exemplary only and in no way foreclose AGIS 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 ZTE'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, AGIS reserves the right to supplement, correct, modify, and/or amend these contentions once such additional information is made available to AGIS. Furthermore, AGIS reserves the right to supplement, correct, modify, and/or amend 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 ZTE, 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.

US9467838	Exemplary Supporting Evidence Regarding Accused Products
1[P]. A computer-implemented method	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of: a computer-implemented method [of claim 1].

US9467838	Exemplary Supporting Evidence Regarding Accused Products
comprising:	The Accused Products meet the claim limitations by providing device-location tracking features such as those features described below. For example, the Accused Products meet the claim limitations because they are pre-installed with Android mobile operating systems containing code for providing device-location tracking features as provided in the claims limitations herein. For example, the Accused Products run applications and/or software that run within the Android mobile operating system and that use components of the Android mobile operating system to provide device-location tracking features. Upon information and belief, in addition to the components and features of the Android mobile operating system itself, the following applications and/or software run within the Android mobile operating system and use components of the Android mobile operating system to provide device-location tracking features: Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Messages, Android Messenger, Google Allo, Google Duo, and Google Chrome. The Accused Products meet the claim limitations under at least two exemplary methods and/or systems, as shown below, which may be useable together or separately. **Find My Device:** Android Device Manager is the predecessor to Find My Device and has been available as a standard, pre-installed feature since 2013 and downloadable as a software application. The current iteration, Find My Device, often called the "new and improved Android Device Manager" or "rebranded Android Device Manager" is now part of the standard Google Play Protect suite which is "built in and enabled on all devices," *i.e.*, the Accused Products running Android OS. Upon information and belief, the Find My Device method also uses and/or works in conjunction with functionalities associated with Google Maps, Google Messages, Android Messenger, Location Access, Google Chrome, and other features which come pre-installed on the Accused Products. For the purposes of avoiding needlessly presenting cumulative and duplicative evidence, AGIS sets forth the Find My Device feature of the Accused Products as representative of this first exemplary method. AGIS reserves the right to supplement these contentions to the extent that defendant requires additional information in accordance with P.R. 3-1 and for any other reason for which it may deem necessary. *See*, *e.g.*, https://www.androidcentral.com/find-my-device; https://support.google.com/android/answer/6160491?hl=en; https://android.googleblog.com/2013/08/find-your-lost-phone-with-android.html;

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	https://play.google.com/store/apps/details?id=com.google.android.apps.adm&hl=en; https://www.blog.google/products/android/google-play-protect/ **Google Maps Share Location:** Share Location is currently included as a standard feature on the Accused Devices operating as a feature of Google Maps. Google Maps is a pre-installed software application in Android OS. The Accused Devices have included the Share Location functionalities since 2009 as part of Google Latitude which was an opt-in feature for Google Maps on Android OS-based mobile devices, such as the Accused Products. Share Location functionalities were briefly shifted from Latitude for Google Maps to Google Plus and Google Hangouts, until reappearing as a standard feature in Google Maps. Upon information and belief, the Share Location method also uses and/or works in conjunction with functionalities associated with Google Maps, Google Messages, Android Messenger, Location Access, Google Allo, Google Duo, Google Chrome, and other features which come pre-installed on the Accused Products. For the purposes of avoiding needlessly presenting cumulative and duplicative evidence, AGIS sets forth Google Maps' Share Location feature of the Accused Products as representative of this second exemplary method. AGIS reserves the right to supplement these contentions to the extent that defendant requires additional information in accordance with P.R. 3-1 and for any other reason for which it may deem necessary. *See*, *e.g.*, https://techcrunch.com/2017/03/22/google-maps-now-lets-you-share-your-location-with-friends-and-family-for-a-specific-period-of-time/; https://googleblog.blogspot.com/2009/02/see-where-your-friends-are-with-google.html; https://googleblog.blogspot.jp/2013/07/a-new-google-maps-app-for-smartphones.html; http://googleplusproject.blogspot.com/2013/05/google-for-android-42.html; https://googleblog.blogspot.com/2013/10/google-hangouts-and-photos-save-some.html

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

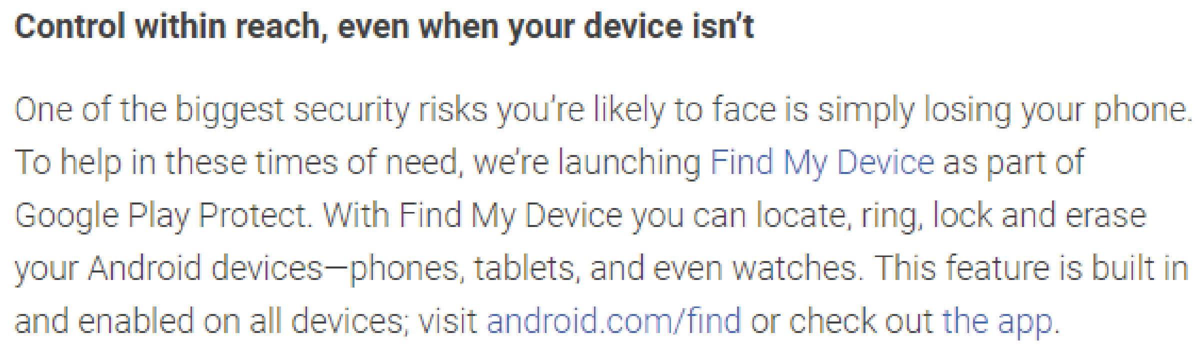
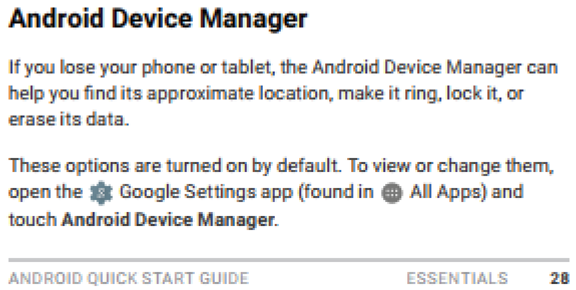
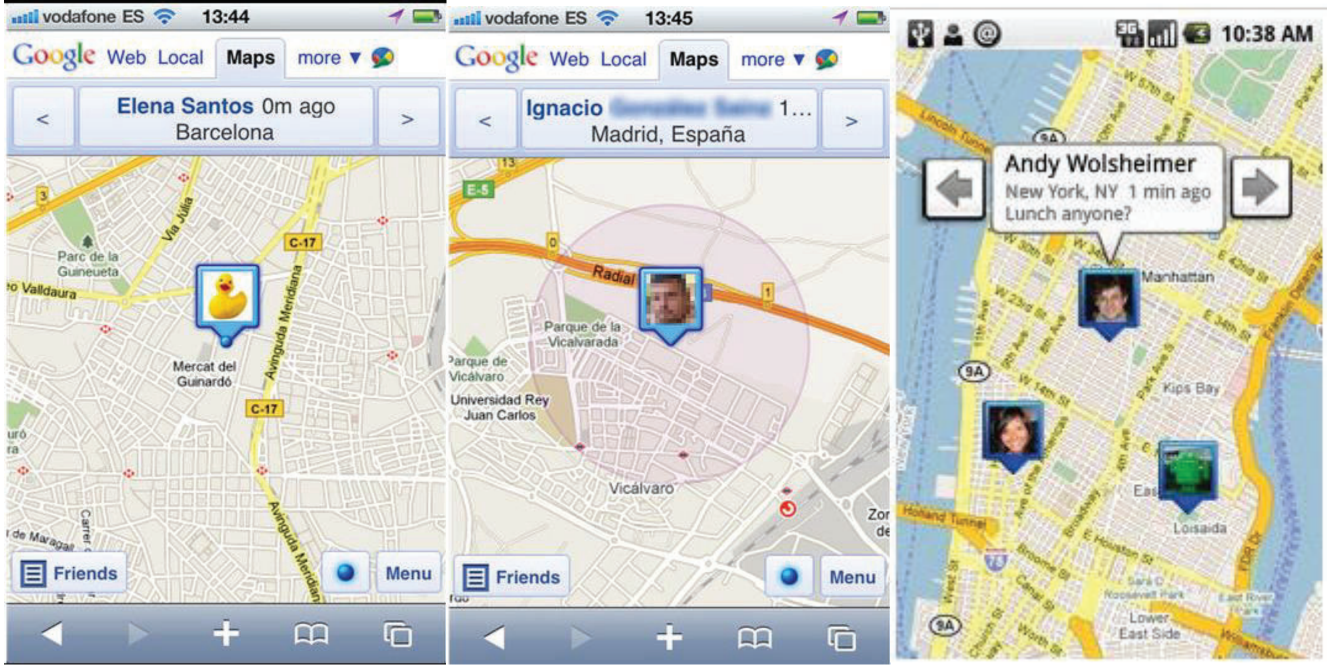
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 **Control within reach, even when your device isn't** One of the biggest security risks you're likely to face is simply losing your phone. To help in these times of need, we're launching Find My Device as part of Google Play Protect. With Find My Device you can locate, ring, lock and erase your Android devices—phones, tablets, and even watches. This feature is built in and enabled on all devices; visit android.com/find or check out the app. *See*, *e.g.*, https://www.blog.google/products/android/google-play-protect/  **Android Device Manager** If you lose your phone or tablet, the Android Device Manager can help you find its approximate location, make it ring, lock it, or erase its data. These options are turned on by default. To view or change them, open the ⚙ Google Settings app (found in ▦ All Apps) and touch **Android Device Manager**. ANDROID QUICK START GUIDE ESSENTIALS 28 https://static.googleusercontent.com/media/www.google.com/en/us/help/hc/images/android/android_ug_50/Android-Lollipop-Quick-Start-Guide.pdf As discussed above and shown below, Google Latitude provided similar features to Google Maps' location sharing feature since 2009:

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 vodafone ES 13:44 Google Web Local Maps more Elena Santos 0m ago Barcelona Parc de la Guineueta Valldaura Via Julia C-17 Avinguda Meridiana Mercat del Guinardó Friends Menu vodafone ES 13:45 Google Web Local Maps more Ignacio [redacted] 1... Madrid, España E-5 Radial Parque de la Vicalvarada Parque de Vicálvaro Universidad Rey Juan Carlos Vicálvaro Friends Menu 10:38 AM Andy Wolsheimer New York, NY 1 min ago Lunch anyone? Manhattan Kips Bay Loisaida Lower East Side Holland Tunnel Lincoln Tunnel http://googlemobile.blogspot.com/2009/02/locate-your-friends-in-real-time-with.html
[1A] performing, by a first device:	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance, by a first device [the operations of claim 1]. For example, the ZTE products, such as Android-based phones and tablets run device-location tracking features and/or software such as Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Messages, and Android Messenger, Google Allo, Google Duo, and Google Chrome.
[1B] joining a communication network	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of: joining a communication network corresponding to a group, wherein joining the

US9467838	Exemplary Supporting Evidence Regarding Accused Products
corresponding to a group, wherein joining the communication network comprises transmitting a message including an identifier corresponding to the group;	communication network comprises transmitting a message including an identifier corresponding to the group. **Regarding Find My Device** and Android Device Manager, the Accused Products require a user to join the corresponding network by: signing-in to the device with an identifier (e.g., Google Account) or linking the device to the identifier by remote means. The sign-in process takes place within the Find My Device software on the Accused Product; by navigating to android.com/find within the Google Chrome browser on the Accused Product; by navigating to google.com and typing "find my phone" within the Google Chrome browser on the Accused Product; or by searching "find my phone" within the search bar of the Google home screen and/or Google application on the Accused Product. Alternatively, the sign-in process may partially or completely take place using credentials already provided when the user associates a Google Account with the Accused Product, e.g., during initial setup of the Accused Product. Subject to discovery, one or more additional or substitute identifiers may to correspond to the group. The sign-in process involves a user entering its Google Account and additional authentication data on the interface of the Accused Product and sending a message containing the Google Account and additional authentication data over a network to the group. The group comprises the multiple devices linked to the identifier. **Regarding Google Maps**, Google Latitude, Google Plus, Google Hangouts, Google Messages, and Android Messenger, Google Duo, Google Allo, and Google Chrome, the Accused Products require a user to join the corresponding network by signing-in to the device with an identifier (e.g., Google Account). The sign-in process takes place within the Google Maps software on the Accused Product or by navigating to maps.google.com within the Google Chrome browser on the Accused Product. Alternatively, the sign-in process may partially or completely take place using credentials already provided when the user associates a Google Account with the Accused Product, e.g., during initial setup of the Accused Product. Subject to discovery, one or more additional or substitute identifiers may to correspond to the group. The sign-in process involves a user entering its Google Account and additional authentication data on the interface of the Accused Product and sending a message containing the Google Account and additional authentication data over a network to the group. The group comprises the multiple identifiers, individuals, profiles, and/or devices associated with the group. Further regarding Google Maps, Google Latitude, Google Plus, Google Hangouts, Google Messages, and Android Messenger, Google Duo, Google Allo, and Google Chrome, the Accused Products alternatively

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	require a user to send a request containing the identifier (e.g., Google Account). Subject to discovery, additional identifiers may be assigned or used to correspond to the group. The request may be an invitation or message that associates a Google Account with one or more Google Accounts for the purposes of sharing locations within the group. The group comprises the multiple identifiers, individuals, profiles, and/or devices associated with the group **<u>Exemplary Support for Find My Device</u>**:

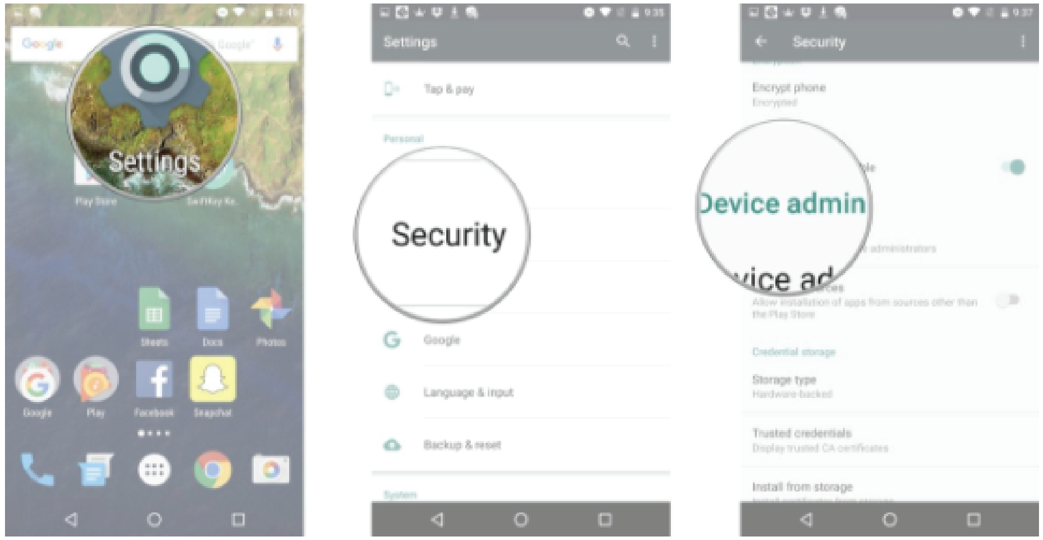
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to enable Find My Device on your phone In newer Android phones, the Find My Device service is already located conveniently in your Settings app, but if you can't find it you can always download Find My Device from the Google Play Store. This locating service has essentially amalgamated with Google to make finding your phone easier. There are just a couple of things you'll need to activate. 1. Launch **Settings**. 2. Tap **Security**. 3. Tap **Device Administration**.  4. Tap **Find My Device** so that a checkmark appears in the checkbox. https://www.androidcentral.com/how-track-android-phone

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

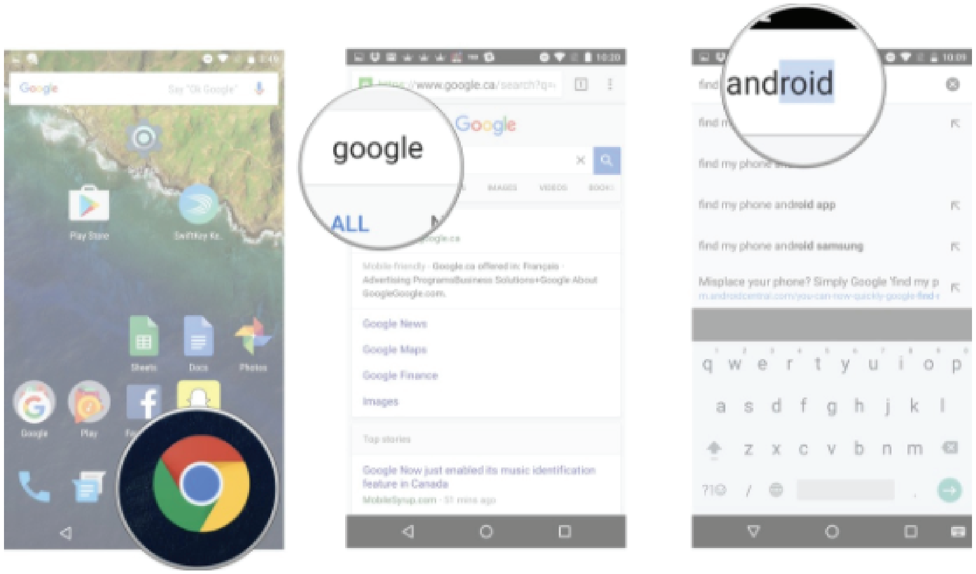
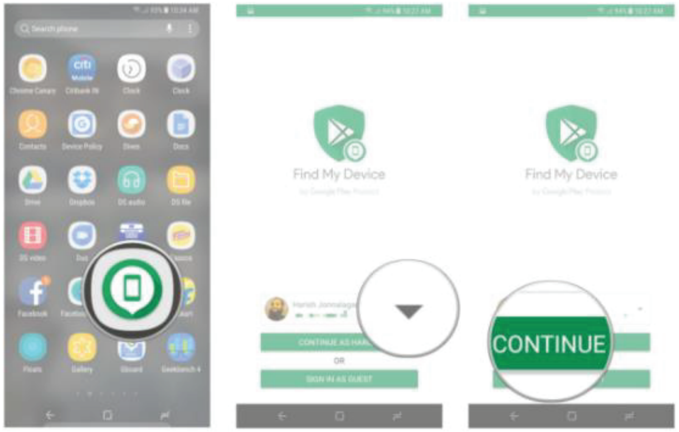
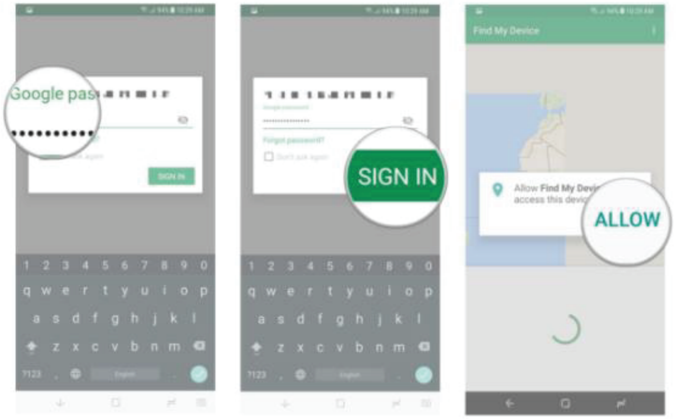
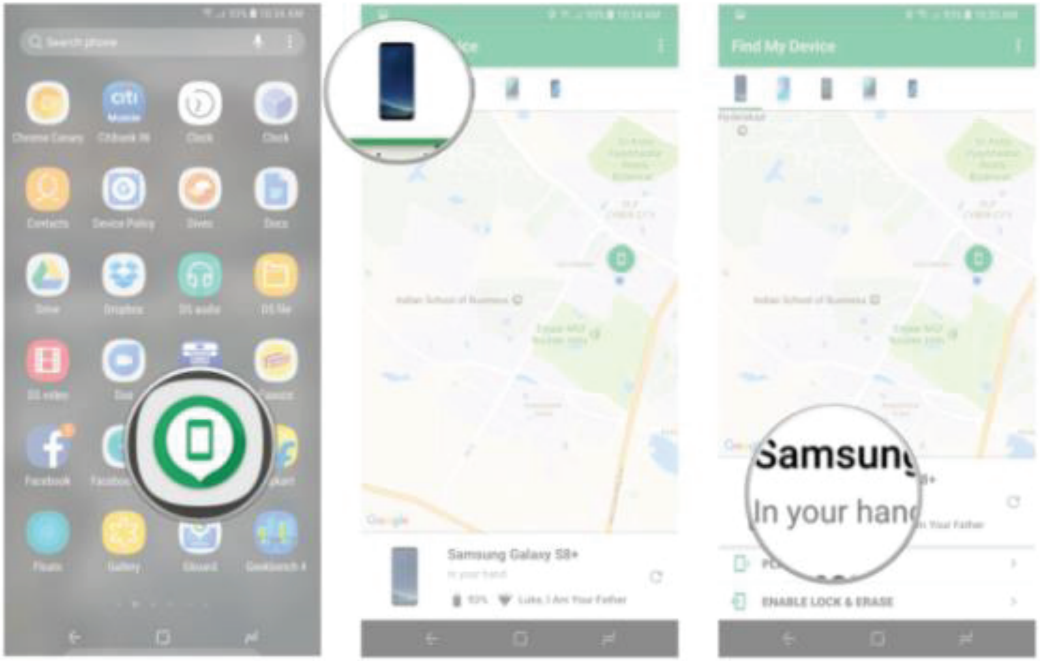
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to locate your phone with Google Should you happen to lose your phone, you can locate its whereabouts by logging into your Google account from any computer or even from another phone. 1. Launch a **web browser** from a phone, tablet, or computer. 2. Navigate to **Google** if it is not your default search engine or home page. 3. Type **find my phone android** in the Google search bar.  4. Tap on **Find My Device** (usually the first option in the search). 5. Enter your **email address** and **password** just as though you were checking your email. If you have 2-step verification set up on your Google account (and you most certainly should), you'll need to complete that process as well. https://www.androidcentral.com/how-track-android-phone

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	1. Open **Find Device** from your home screen or app drawer. 2. Select the **Google account** you want to use the service with. 3. Hit the **Continue as** button.  Find My Device CONTINUE AS HARISH OR SIGN IN AS GUEST CONTINUE 4. Enter your **Google account password**. 5. Tap **Sign in**. 6. Give **location access** to the service.  Google pas SIGN IN Find My Device Allow Find My Devi access this devic ALLOW https://www.androidcentral.com/find-my-device

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	Once you're signed in to Find My Device, you'll see a map with your current location as well as the make and model of your phone, and two options — Play Sound, and Enable Lock & Erase. Hitting the latter option will allow you to start using the Lock and Erase functions. If you've signed into more than one phone, you can select a particular device by browsing the list at the top of the screen. 1. Open **Find My Device** from your home screen or app drawer. 2. Select your phone from the **list of devices at the top of the screen**. 3. See if your phone is **discoverable**.  Find My Device Samsung Galaxy S8+ Samsung In your hand ENABLE LOCK & ERASE https://www.androidcentral.com/find-my-device

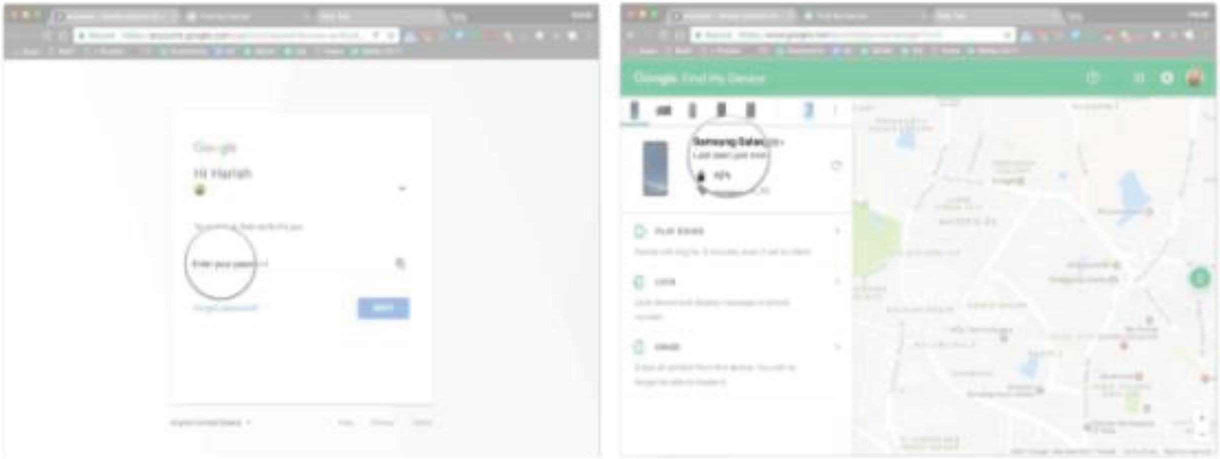
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to locate your phone over the internet If you've lost your phone, you can remotely locate it through the Find My Device website. You'll need to sign in to the Google account that was used to set up Find My Device. It takes a few seconds, but the service should be able to track your phone. Alternatively, you can also do a Google search for "find my phone" to locate your handset. 1. Head to the **Find My Device website**. 2. Sign in to your **Google account**. 3. Check if your device is **visible**.  https://www.androidcentral.com/find-my-device **<u>Exemplary Support for Google Maps</u>**:

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

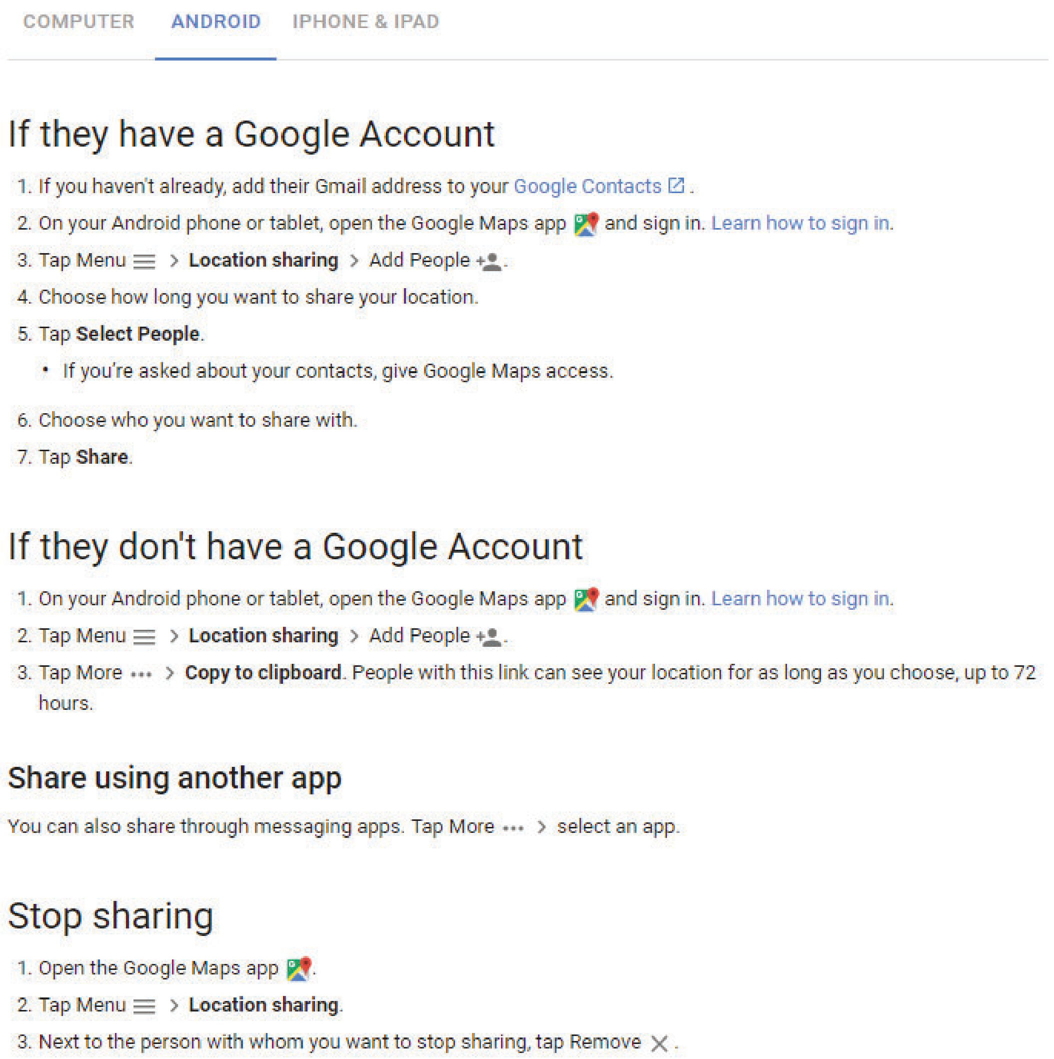
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 If they have a Google Account 1. If you haven't already, add their Gmail address to your Google Contacts. 2. On your Android phone or tablet, open the Google Maps app and sign in. Learn how to sign in. 3. Tap Menu ☰ > **Location sharing** > Add People. 4. Choose how long you want to share your location. 5. Tap **Select People**. - If you're asked about your contacts, give Google Maps access. 6. Choose who you want to share with. 7. Tap **Share**. If they don't have a Google Account 1. On your Android phone or tablet, open the Google Maps app and sign in. Learn how to sign in. 2. Tap Menu ☰ > **Location sharing** > Add People. 3. Tap More ••• > **Copy to clipboard**. People with this link can see your location for as long as you choose, up to 72 hours. Share using another app You can also share through messaging apps. Tap More ••• > select an app. Stop sharing 1. Open the Google Maps app. 2. Tap Menu ☰ > **Location sharing**. 3. Next to the person with whom you want to stop sharing, tap Remove ✕. https://support.google.com/maps/answer/7326816?co=GENIE.Platform%3DAndroid&oco=1

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

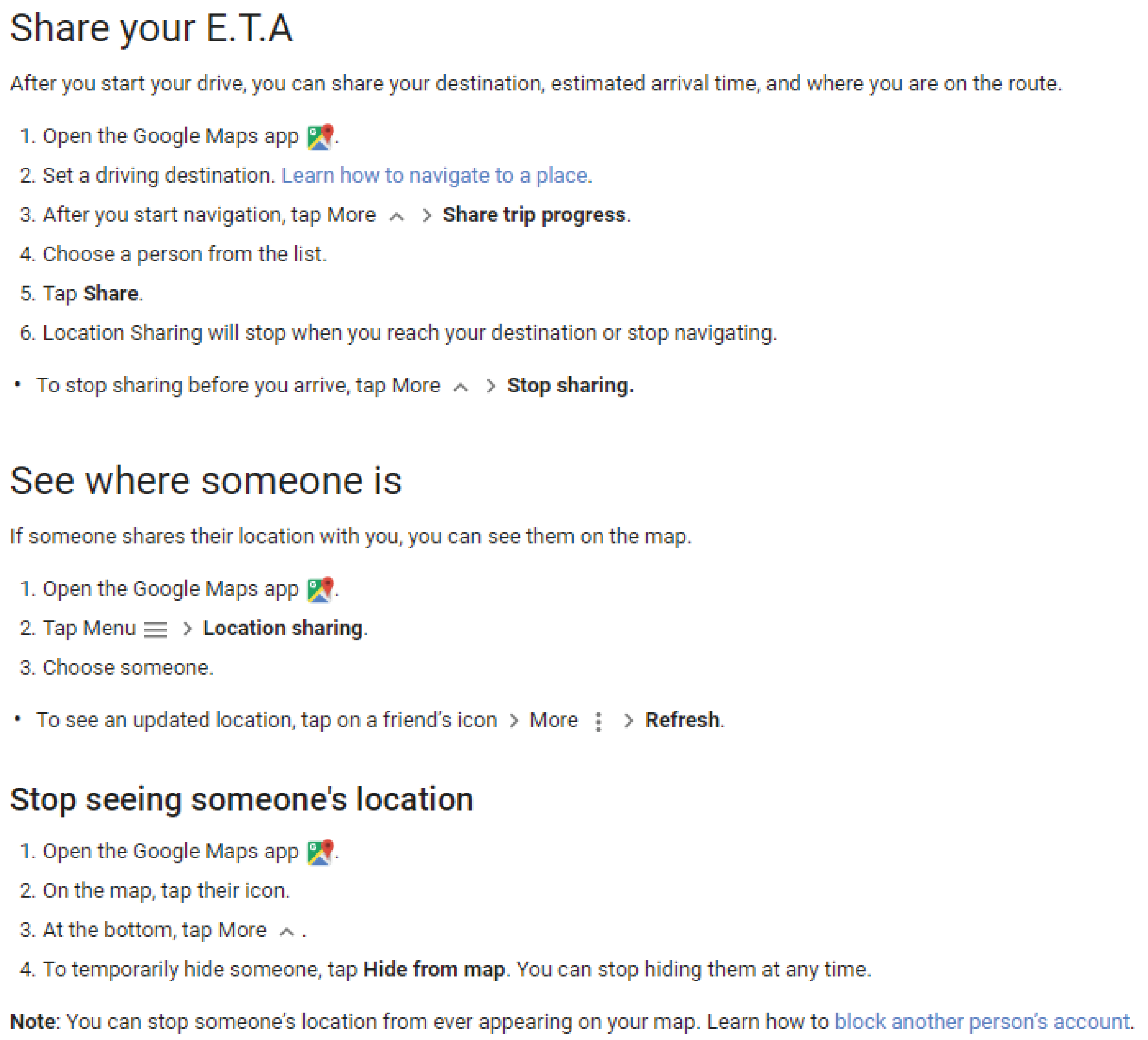
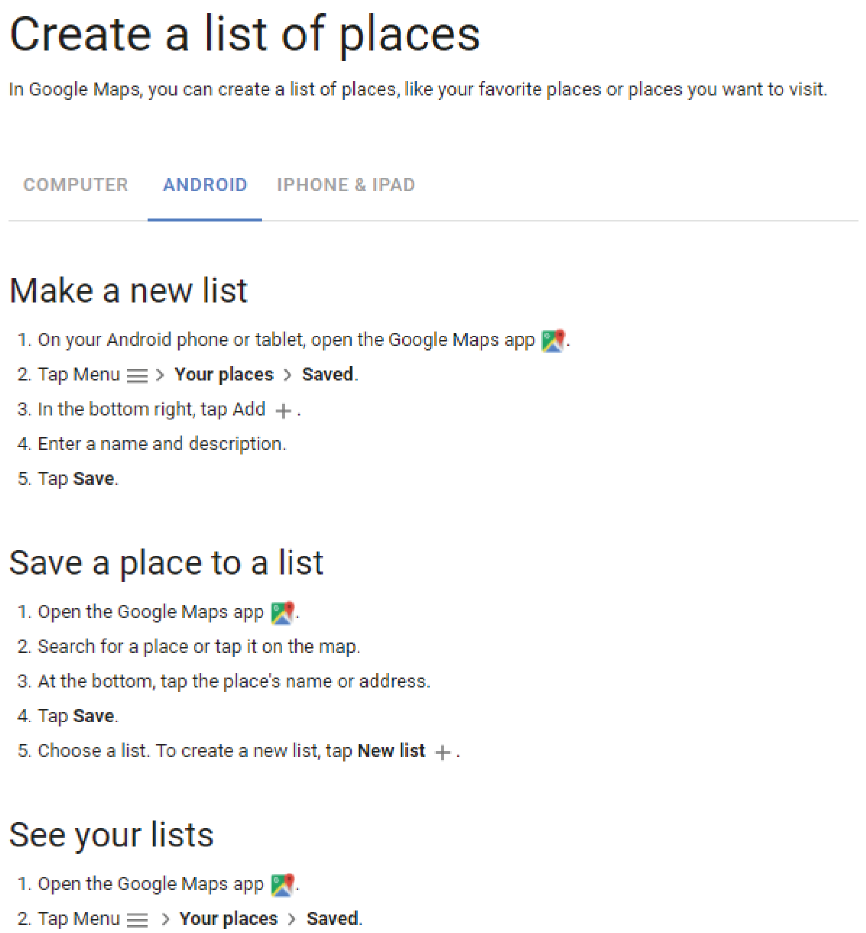
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Share your E.T.A After you start your drive, you can share your destination, estimated arrival time, and where you are on the route. 1. Open the Google Maps app . 2. Set a driving destination. Learn how to navigate to a place. 3. After you start navigation, tap More ^ > **Share trip progress**. 4. Choose a person from the list. 5. Tap **Share**. 6. Location Sharing will stop when you reach your destination or stop navigating. • To stop sharing before you arrive, tap More ^ > **Stop sharing.** See where someone is If someone shares their location with you, you can see them on the map. 1. Open the Google Maps app . 2. Tap Menu ☰ > **Location sharing**. 3. Choose someone. • To see an updated location, tap on a friend's icon > More ⋮ > **Refresh**. Stop seeing someone's location 1. Open the Google Maps app . 2. On the map, tap their icon. 3. At the bottom, tap More ^ . 4. To temporarily hide someone, tap **Hide from map**. You can stop hiding them at any time. **Note**: You can stop someone's location from ever appearing on your map. Learn how to block another person's account. https://support.google.com/maps/answer/7326816?co=GENIE.Platform%3DAndroid&oco=1

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Create a list of places In Google Maps, you can create a list of places, like your favorite places or places you want to visit. Make a new list 1. On your Android phone or tablet, open the Google Maps app. 2. Tap Menu ☰ > **Your places** > **Saved**. 3. In the bottom right, tap Add +. 4. Enter a name and description. 5. Tap **Save**. Save a place to a list 1. Open the Google Maps app. 2. Search for a place or tap it on the map. 3. At the bottom, tap the place's name or address. 4. Tap **Save**. 5. Choose a list. To create a new list, tap **New list** +. See your lists 1. Open the Google Maps app. 2. Tap Menu ☰ > **Your places** > **Saved**. https://support.google.com/maps/answer/7280933?hl=en&ref_topic=7301134&co=GENIE.Platform%3DAndroid&oco=1

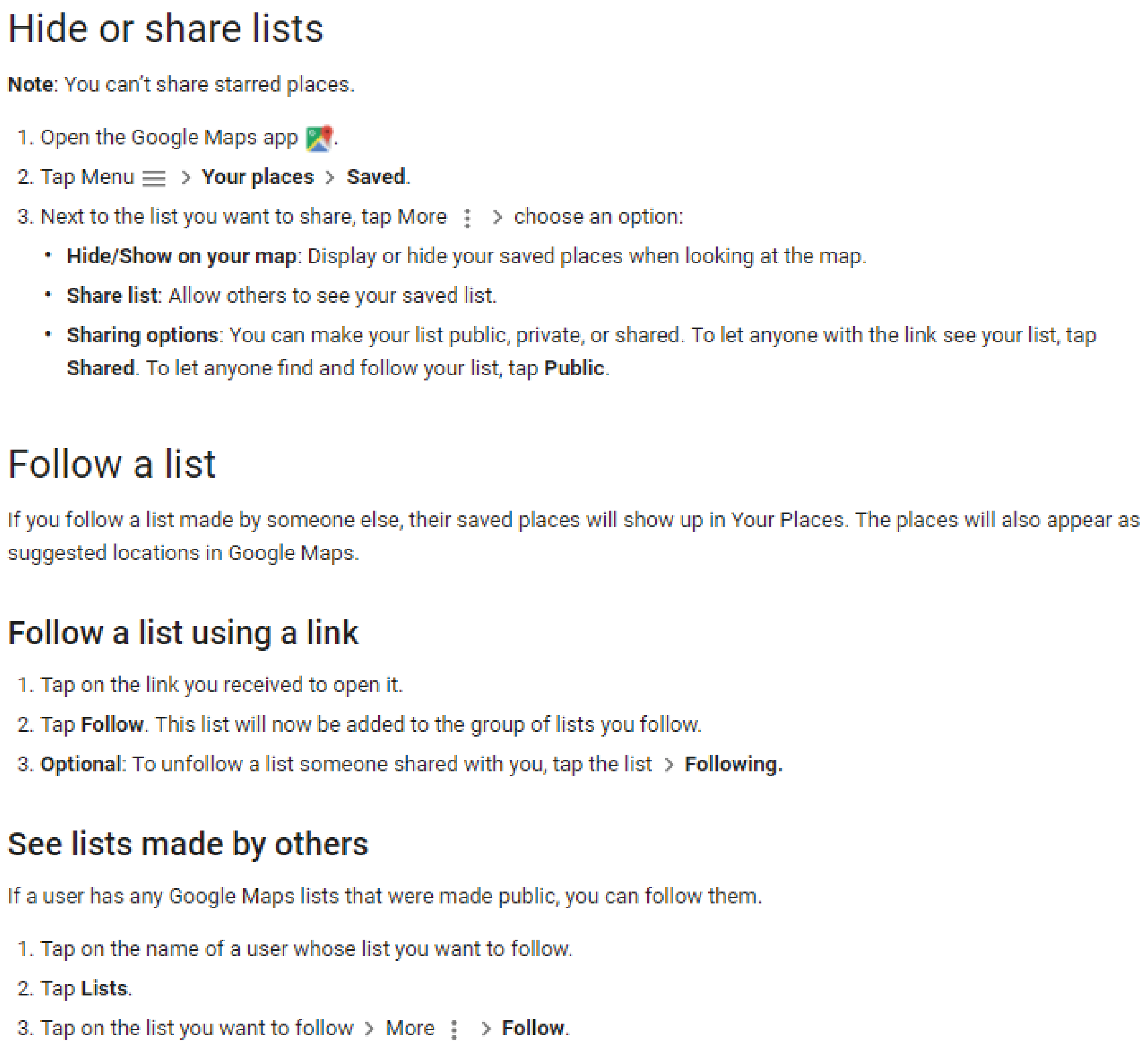
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Hide or share lists **Note**: You can't share starred places. 1. Open the Google Maps app. 2. Tap Menu ☰ > **Your places** > **Saved**. 3. Next to the list you want to share, tap More ⋮ > choose an option: • **Hide/Show on your map**: Display or hide your saved places when looking at the map. • **Share list**: Allow others to see your saved list. • **Sharing options**: You can make your list public, private, or shared. To let anyone with the link see your list, tap **Shared**. To let anyone find and follow your list, tap **Public**. Follow a list If you follow a list made by someone else, their saved places will show up in Your Places. The places will also appear as suggested locations in Google Maps. Follow a list using a link 1. Tap on the link you received to open it. 2. Tap **Follow**. This list will now be added to the group of lists you follow. 3. **Optional**: To unfollow a list someone shared with you, tap the list > **Following.** See lists made by others If a user has any Google Maps lists that were made public, you can follow them. 1. Tap on the name of a user whose list you want to follow. 2. Tap **Lists**. 3. Tap on the list you want to follow > More ⋮ > **Follow**. https://support.google.com/maps/answer/7280933?hl=en&ref_topic=7301134&co=GENIE.Platform%3DAndroid&oco=1

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

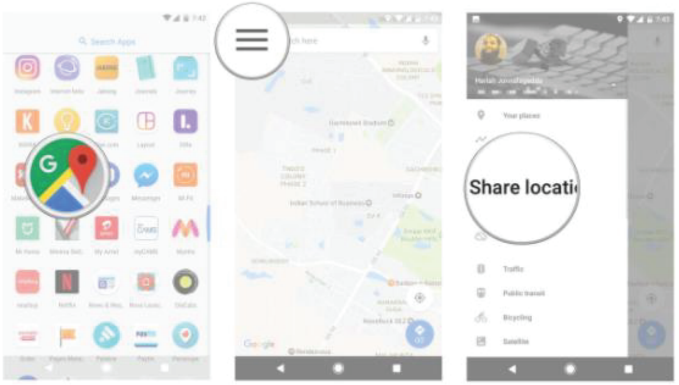
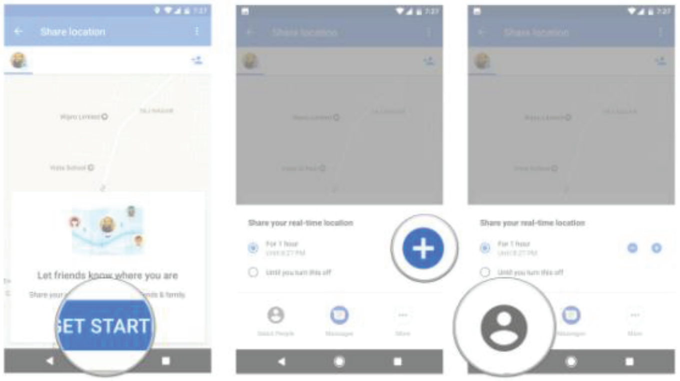
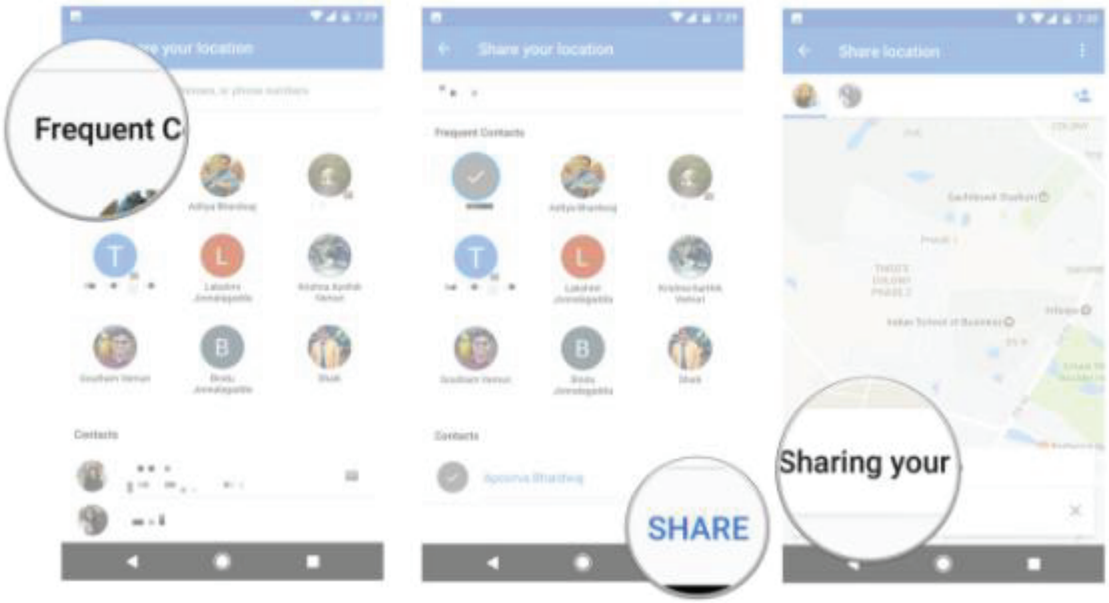
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to share your location in Google Maps 1. Open **Google Maps** from the app drawer or the home screen. 2. Tap the **hamburger menu** (the three horizontal lines) on the top left corner of the screen. 3. Select **Share location**.  4. Tap **Get Started**. 5. Use the **+** icon to select a time period or select the **Until you turn this off** setting to share your location indefinitely. 6. Tap **Select People**.  Share location Let friends know where you are ET START Share your real-time location For 1 hour Until you turn this off https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	7. You'll see a list of your frequent contacts at the top, along with a full list of contacts. **Pick the contacts** by tapping their name. 8. Once you've selected the contacts you want to share your location to, tap **Share**. 9. You'll see a message saying that the selected contact can view your location.  Frequent C Share your location Frequent Contacts Contacts SHARE Share location Sharing your https://www.androidcentral.com/how-share-location-google-maps

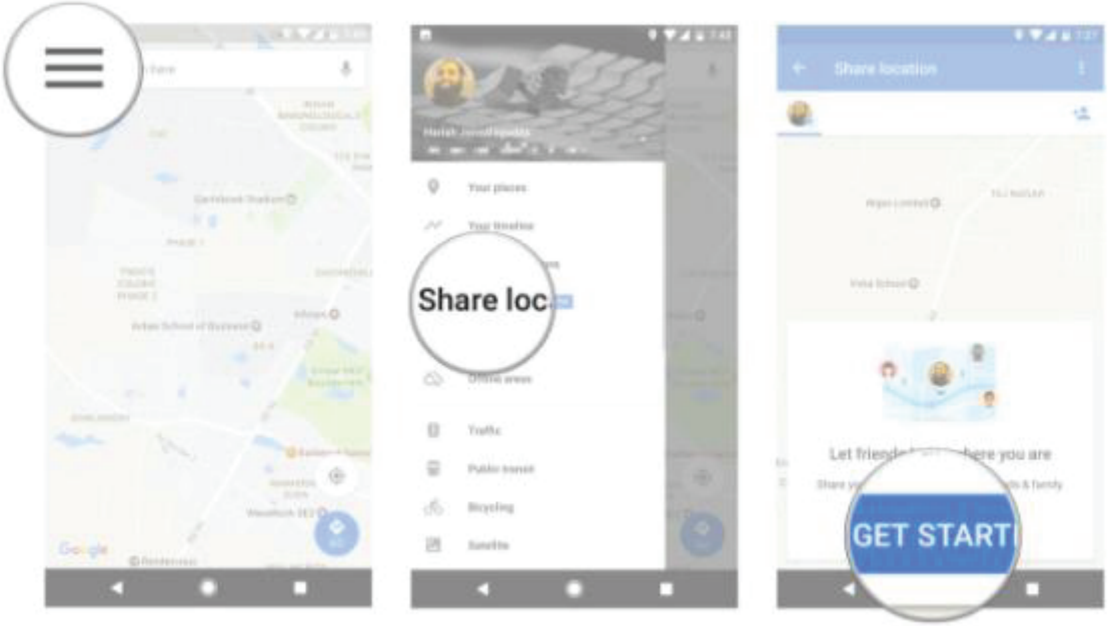
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to create a shareable link You can also create a link and use it to share your location easily. Here's how to do it: 1. Tap the **hamburger menu** on the top left corner of the screen. 2. Select **Share location**. 3. Tap **Get Started**.  Share loc Share location Your places Your timeline Offline areas Traffic Public transit Bicycling Satellite Let friends know where you are GET START https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

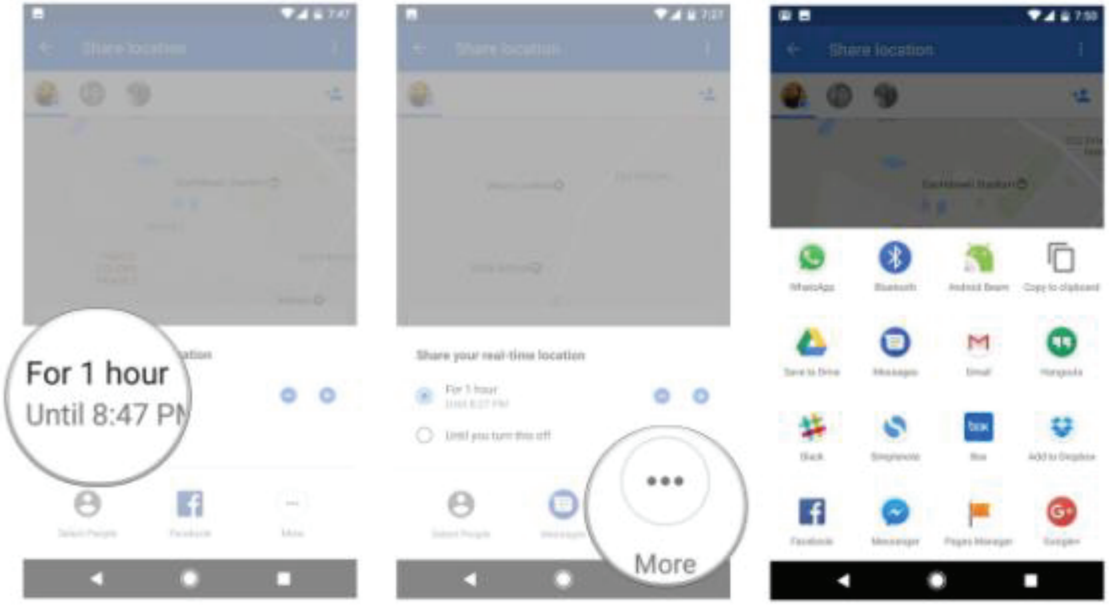
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	4. Select the **amount of time** you want to share your location. 5. Tap **More**. 6. Select **your app of choice** to create and send a unique URL that broadcasts your current location. You can email it, send the link via Messenger, or even tweet it to the intended recipient.  Share location For 1 hour Until 8:47 PM Share your real-time location For 1 hour Until you turn this off More WhatsApp Bluetooth Android Beam Copy to clipboard Save to Drive Messages Email Hangouts Slack Simplenote Box Add to Dropbox Facebook Messenger Pages Manager Google+ https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

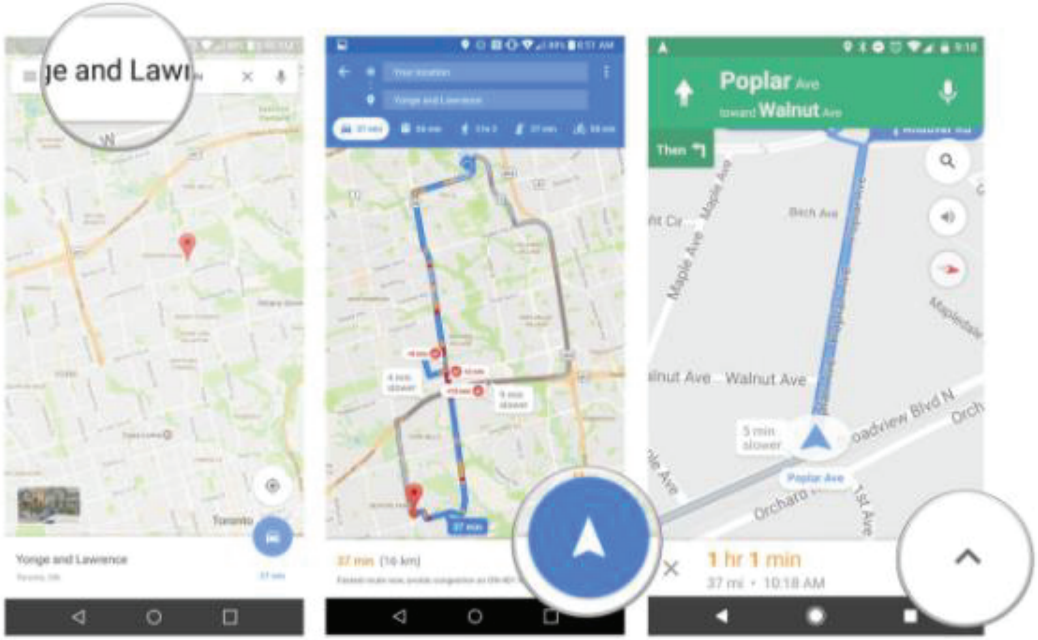
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to share your navigation directions while you walk, drive or transit One of the best ways to share your location with someone is during a drive, walk or transit. If you're meeting a friend or family member somewhere, or navigating towards their home, Google Maps lets you share your location with them for the duration of the trip. It's magic! 1. In the **search bar** enter your destination. 2. Pick your navigation type (drive, transit, walk) and press the blue **navigate button**. 3. Tap the arrow next to the **time-to-destination** number at the bottom of the screen.  ge and Lawr Toronto Yonge and Lawrence 37 min (16 km) 37 min Poplar Ave toward Walnut Ave Then Birch Ave Maple Ave Walnut Ave 5 min slower Poplar Ave Orchard 1st Ave 1 hr 1 min 37 mi · 10:18 AM https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

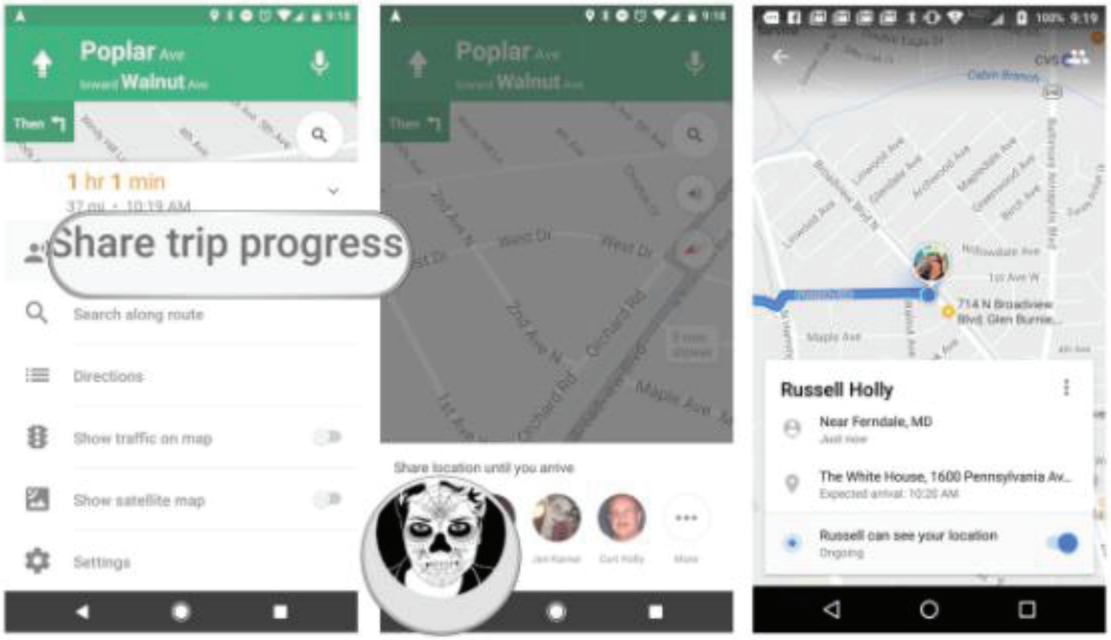
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	4. Tap Share trip progress. 5. Choose one or more contacts to share trip progress.  Poplar Ave toward Walnut Ave Then 1 hr 1 min 37 mi · 10:19 AM Share trip progress Search along route Directions Show traffic on map Show satellite map Settings Share location until you arrive More Russell Holly Near Ferndale, MD Just now The White House, 1600 Pennsylvania Av... Expected arrival: 10:20 AM Russell can see your location Ongoing 714 N Broadview Blvd, Glen Burnie... You can also stop sharing your location with someone before a trip ends. https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

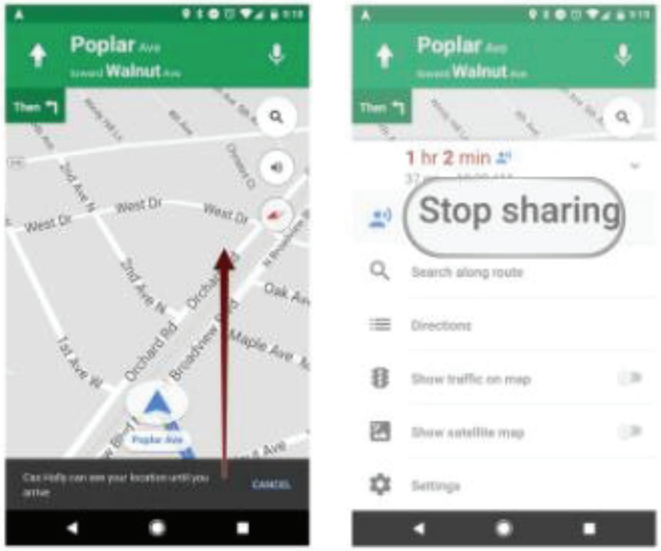
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	1. Tap the arrow next to the **time-to-destination** number at the bottom of the screen. 2. Tap **Stop sharing**.  Poplar Ave toward Walnut Ave Then 2nd Ave N West Dr Orchard Rd Broadview Oak Ave Maple Ave 1st Ave W Poplar Ave CANCEL 1 hr 2 min Stop sharing Search along route Directions Show traffic on map Show satellite map Settings That's it! Are you excited that location sharing is back in Google Maps? How often do you use the feature? https://www.androidcentral.com/how-share-location-google-maps As shown below, a group may also be defined within Google Contacts.

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

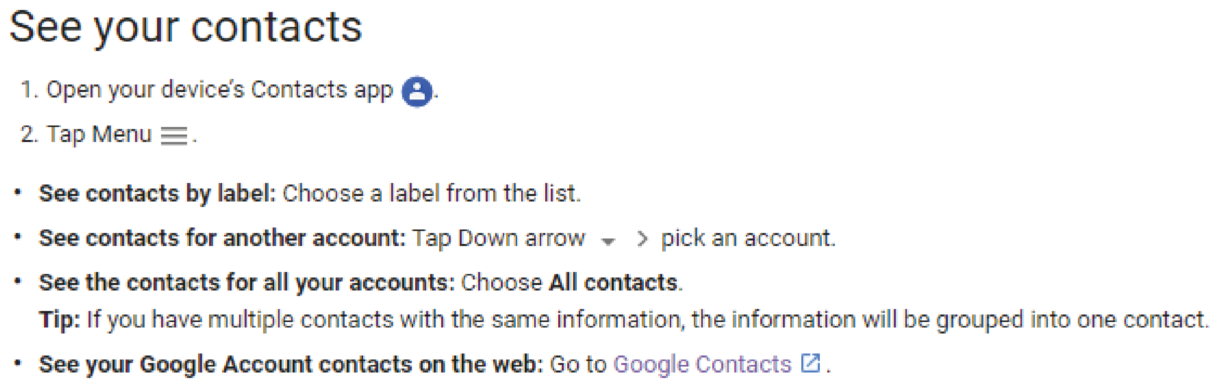
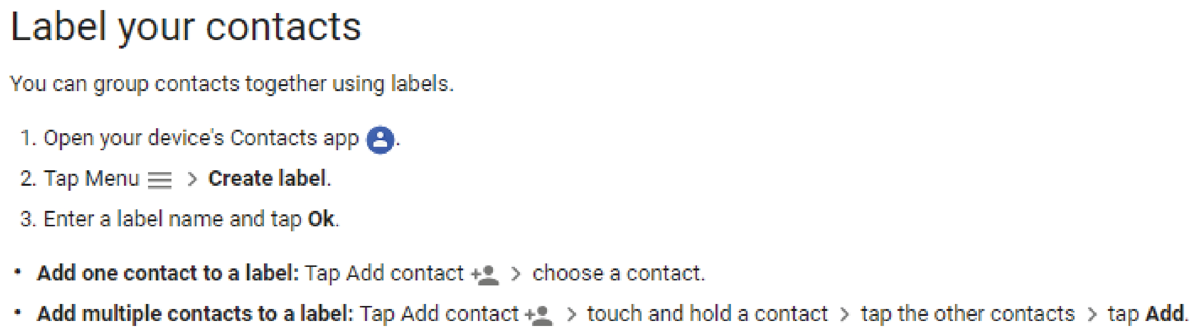
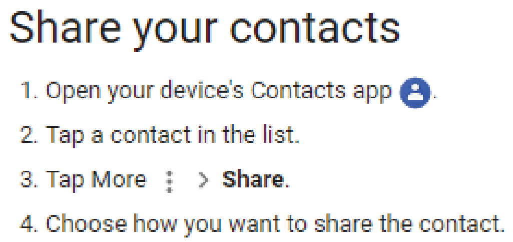
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 **See your contacts** 1. Open your device's Contacts app. 2. Tap Menu ≡. - **See contacts by label:** Choose a label from the list. - **See contacts for another account:** Tap Down arrow ▾ > pick an account. - **See the contacts for all your accounts:** Choose **All contacts**. **Tip:** If you have multiple contacts with the same information, the information will be grouped into one contact. - **See your Google Account contacts on the web:** Go to Google Contacts. https://support.google.com/android/answer/6118731?hl=en&ref_topic=6118711  **Label your contacts** You can group contacts together using labels. 1. Open your device's Contacts app. 2. Tap Menu ≡ > **Create label**. 3. Enter a label name and tap **Ok**. - **Add one contact to a label:** Tap Add contact > choose a contact. - **Add multiple contacts to a label:** Tap Add contact > touch and hold a contact > tap the other contacts > tap **Add**. https://support.google.com/android/answer/6118731?hl=en&ref_topic=6118711  **Share your contacts** 1. Open your device's Contacts app. 2. Tap a contact in the list. 3. Tap More ⋮ > **Share**. 4. Choose how you want to share the contact. https://support.google.com/android/answer/6118731?hl=en&ref_topic=6118711

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

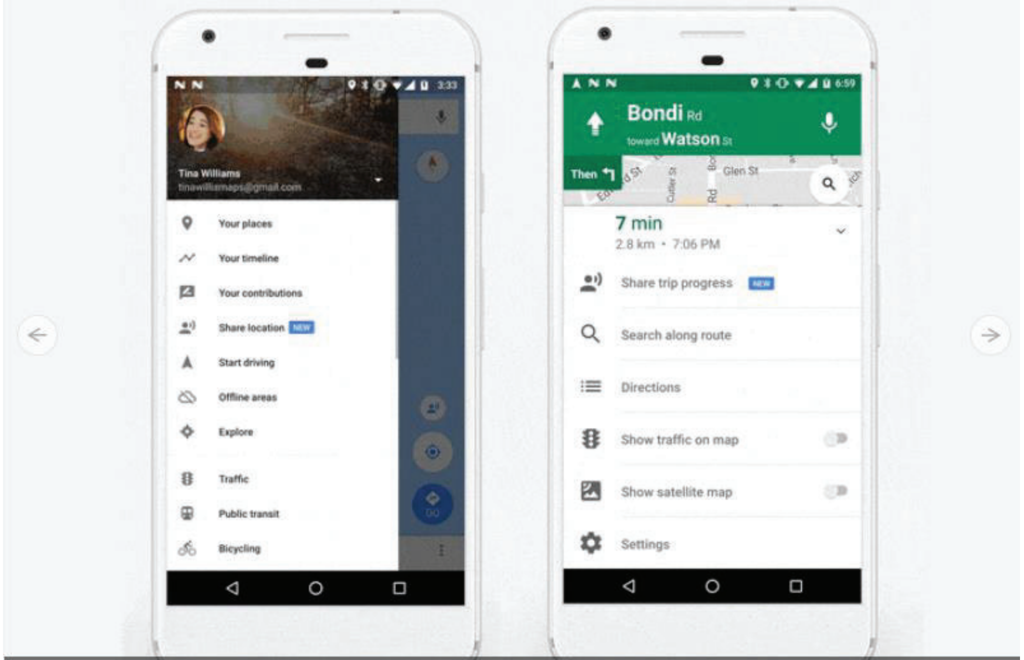
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Tina Williams tinawilliamsmaps@gmail.com Your places Your timeline Your contributions Share location NEW Start driving Offline areas Explore Traffic Public transit Bicycling Bondi Rd toward Watson St Then Glen St 7 min 2.8 km • 7:06 PM Share trip progress NEW Search along route Directions Show traffic on map Show satellite map Settings Ways to start location sharing: (left) from the app navigation drawer, and (right) from the menu, you can drag up while navigating somewhere. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

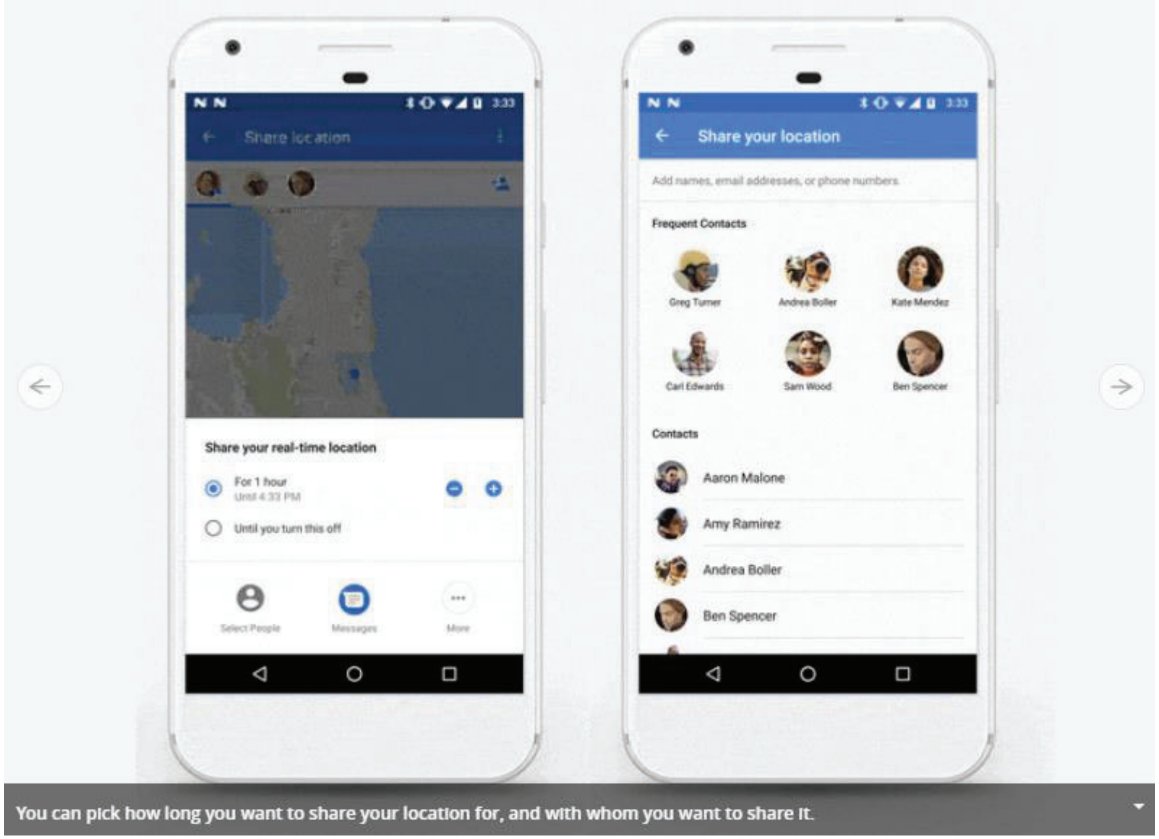
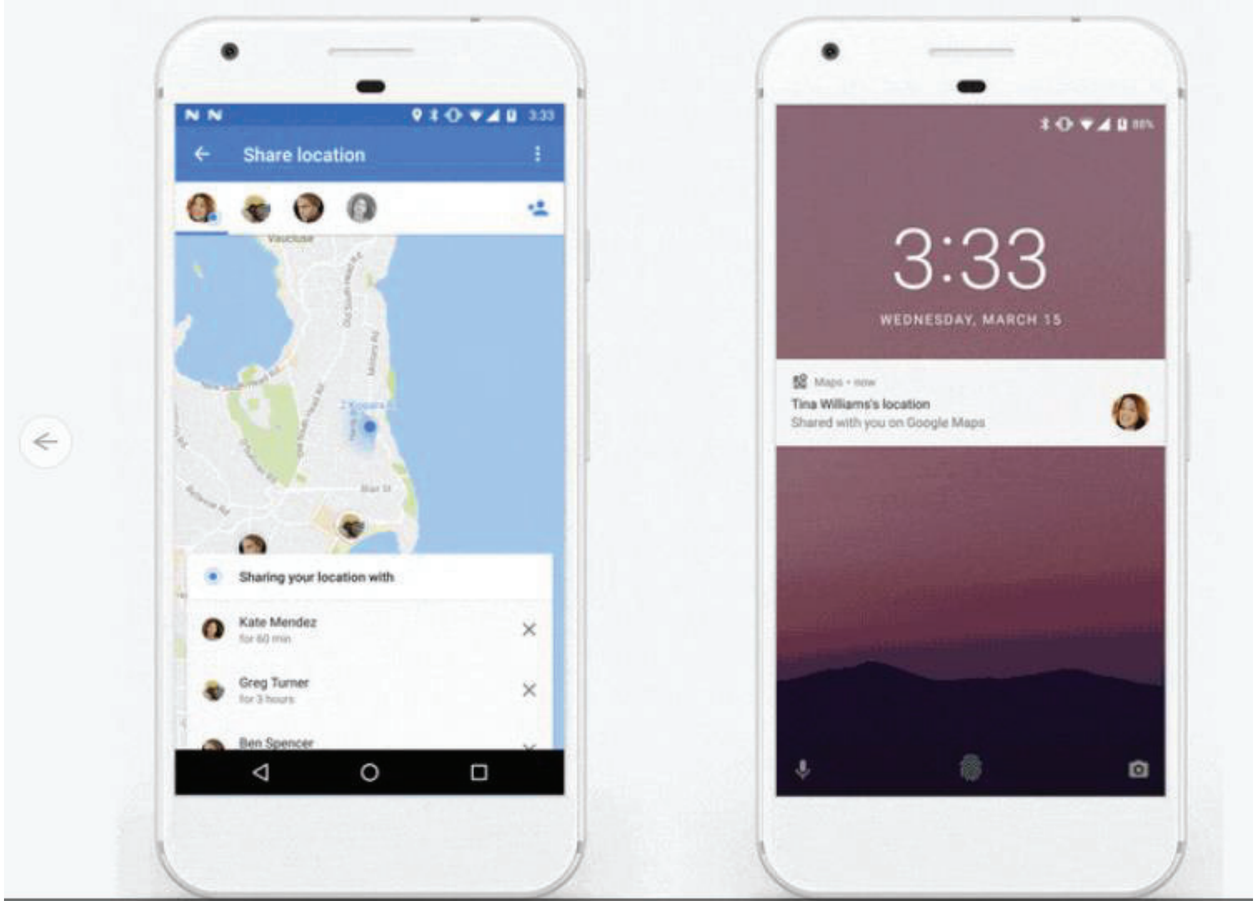
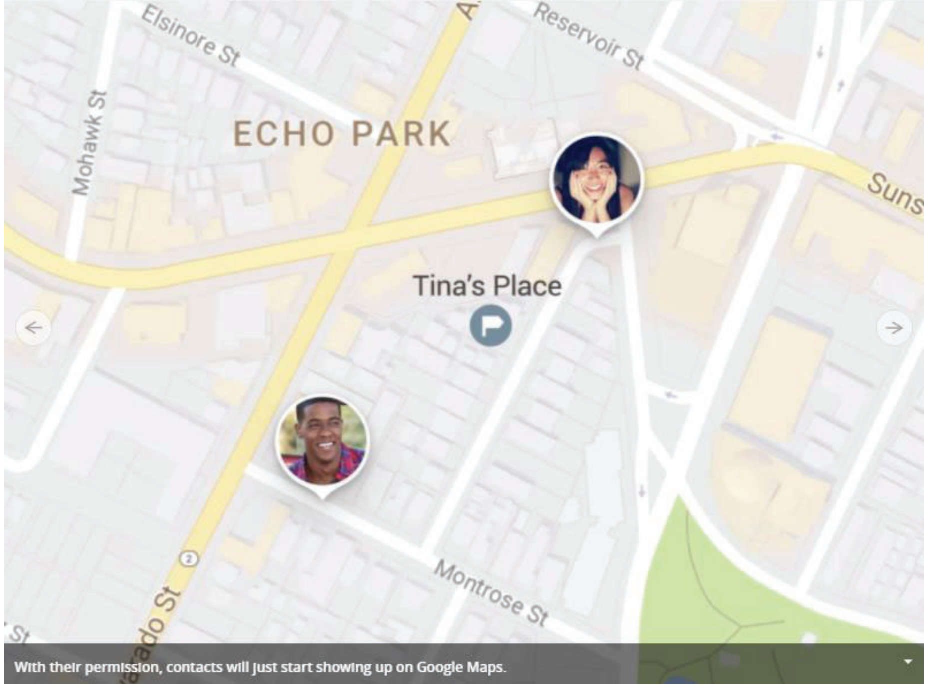
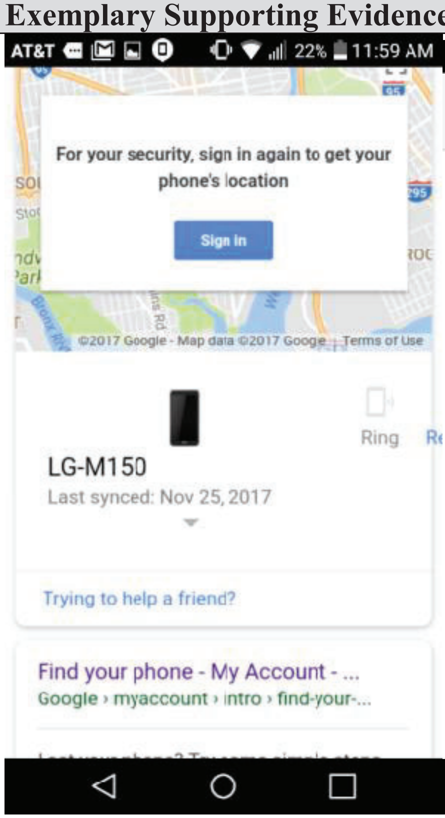
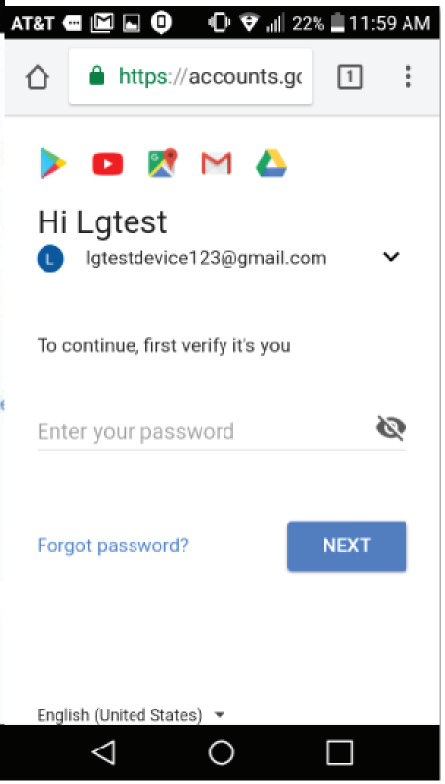
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Share location Share your real-time location For 1 hour Until 4:33 PM Until you turn this off Select People Messages More Share your location Add names, email addresses, or phone numbers Frequent Contacts Greg Turner Andrea Boller Kate Mendez Carl Edwards Sam Wood Ben Spencer Contacts Aaron Malone Amy Ramirez Andrea Boller Ben Spencer You can pick how long you want to share your location for, and with whom you want to share it. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Share location Sharing your location with Kate Mendez for 60 min Greg Turner for 3 hours Ben Spencer 3:33 WEDNESDAY, MARCH 15 Maps • now Tina Williams's location Shared with you on Google Maps People you've shared with will get a notification from Google Maps. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Elsinore St Reservoir St Mohawk St ECHO PARK Suns Tina's Place Montrose St ado St With their permission, contacts will just start showing up on Google Maps. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/ **Exemplary Find My Device Screenshots**:

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 For your security, sign in again to get your phone's location LG-M150 Last synced: Nov 25, 2017 Trying to help a friend? Find your phone - My Account - ... Google › myaccount › intro › find-your-...  https://accounts.gc Hi Lgtest lgtestdevice123@gmail.com To continue, first verify it's you Enter your password Forgot password? English (United States) **<u>Exemplary Google Maps Screenshots</u>:**

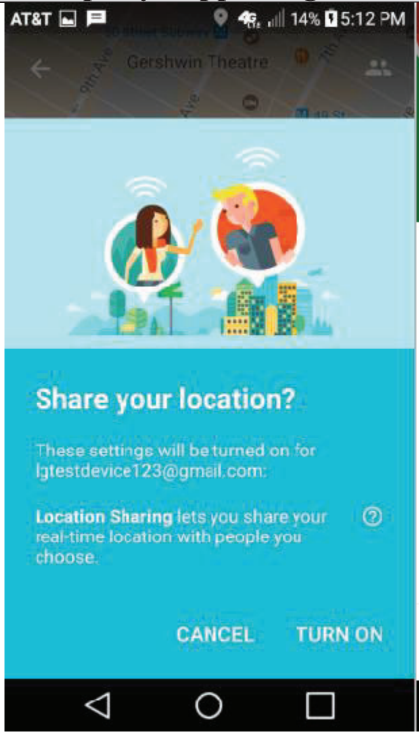
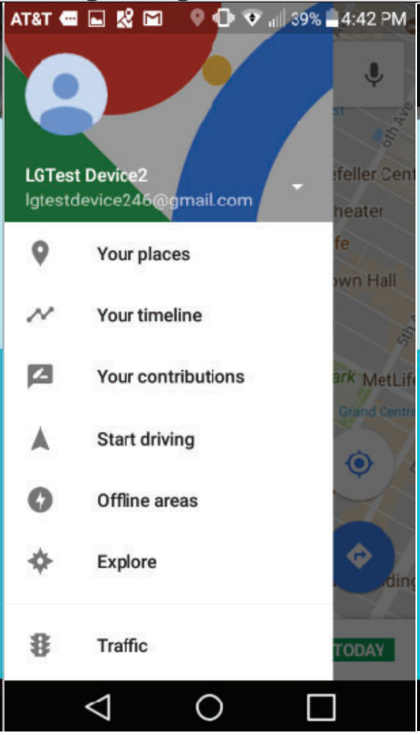
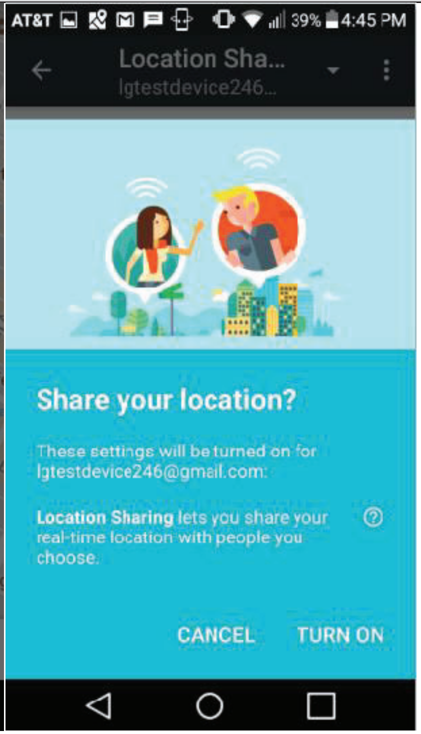
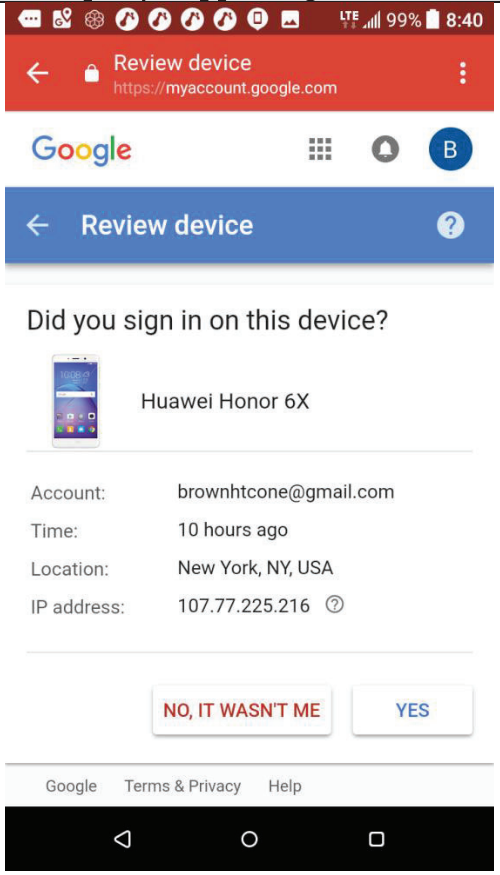
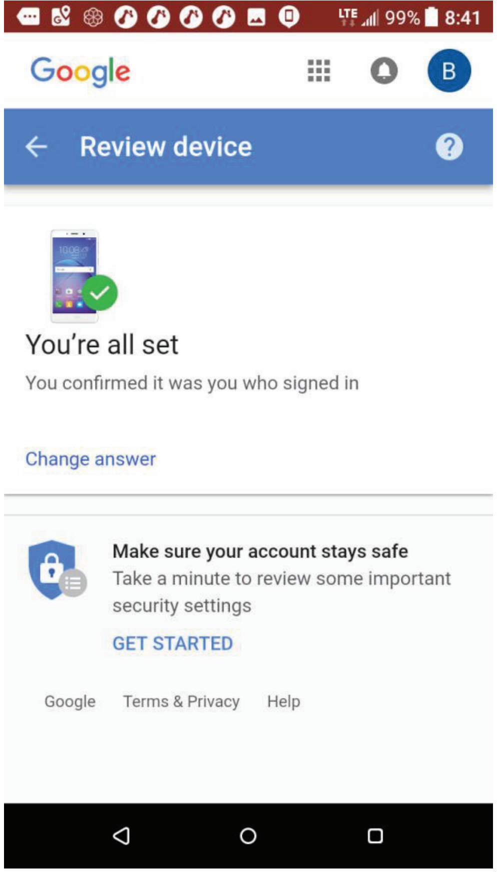
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 14% 5:12 PM Share your location? These settings will be turned on for lgtestdevice123@gmail.com: **Location Sharing** lets you share your real-time location with people you choose. CANCEL TURN ON  AT&T 39% 4:42 PM LGTest Device2 lgtestdevice246@gmail.com Your places Your timeline Your contributions Start driving Offline areas Explore Traffic  AT&T 39% 4:45 PM Location Sha... lgtestdevice246... Share your location? These settings will be turned on for lgtestdevice246@gmail.com: **Location Sharing** lets you share your real-time location with people you choose. CANCEL TURN ON The Accused Products require a user to join the corresponding network by: signing-in to the device with an identifier (e.g., Google Account) or linking the device to the identifier by remote means. The message received by the first device relates to the second device joining into a group with the first device.

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 LTE 99% 8:40 Review device https://myaccount.google.com Google B Review device Did you sign in on this device? Huawei Honor 6X Account: brownhtcone@gmail.com Time: 10 hours ago Location: New York, NY, USA IP address: 107.77.225.216 NO, IT WASN'T ME YES Google Terms & Privacy Help  LTE 99% 8:41 Google B Review device You're all set You confirmed it was you who signed in Change answer Make sure your account stays safe Take a minute to review some important security settings GET STARTED Google Terms & Privacy Help e.g., above, the device receives a message indicating that the second device has joined the group, i.e. the google account. **<u>Exemplary Source Code</u>**:

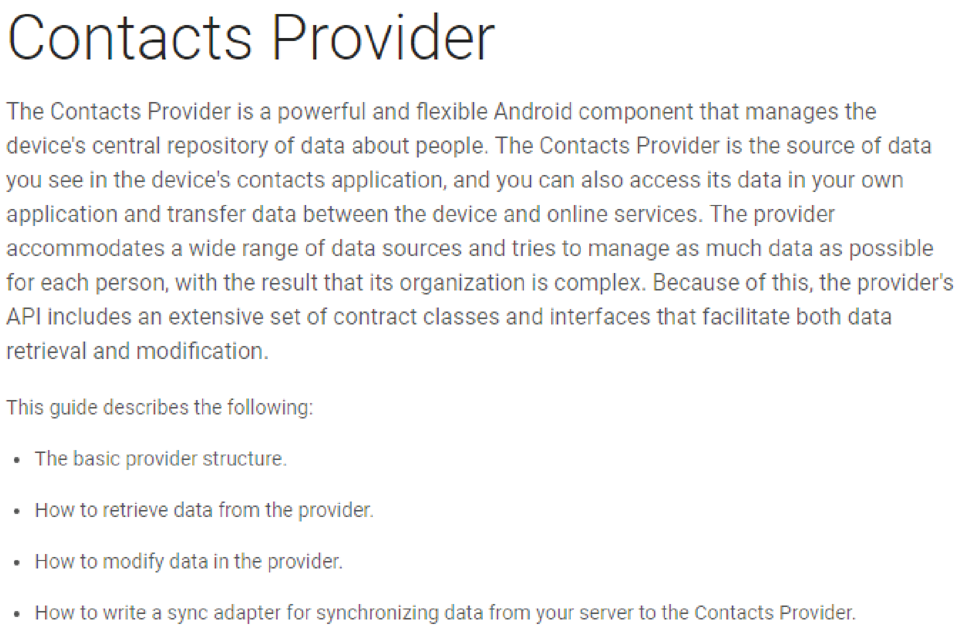
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	The above functionality is performed at least in part by the following publicly available source code and/or source code that invokes or is invoked by the following source code (or a substantially similar copy compiled and loaded onto the Accused Products by ZTE): AGIS reserves the right to amend these contentions to include additional source code as discovery progresses and as additional source code is made available.  Contacts Provider The Contacts Provider is a powerful and flexible Android component that manages the device's central repository of data about people. The Contacts Provider is the source of data you see in the device's contacts application, and you can also access its data in your own application and transfer data between the device and online services. The provider accommodates a wide range of data sources and tries to manage as much data as possible for each person, with the result that its organization is complex. Because of this, the provider's API includes an extensive set of contract classes and interfaces that facilitate both data retrieval and modification. This guide describes the following: • The basic provider structure. • How to retrieve data from the provider. • How to modify data in the provider. • How to write a sync adapter for synchronizing data from your server to the Contacts Provider. https://developer.android.com/guide/topics/providers/contacts-provider.html

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

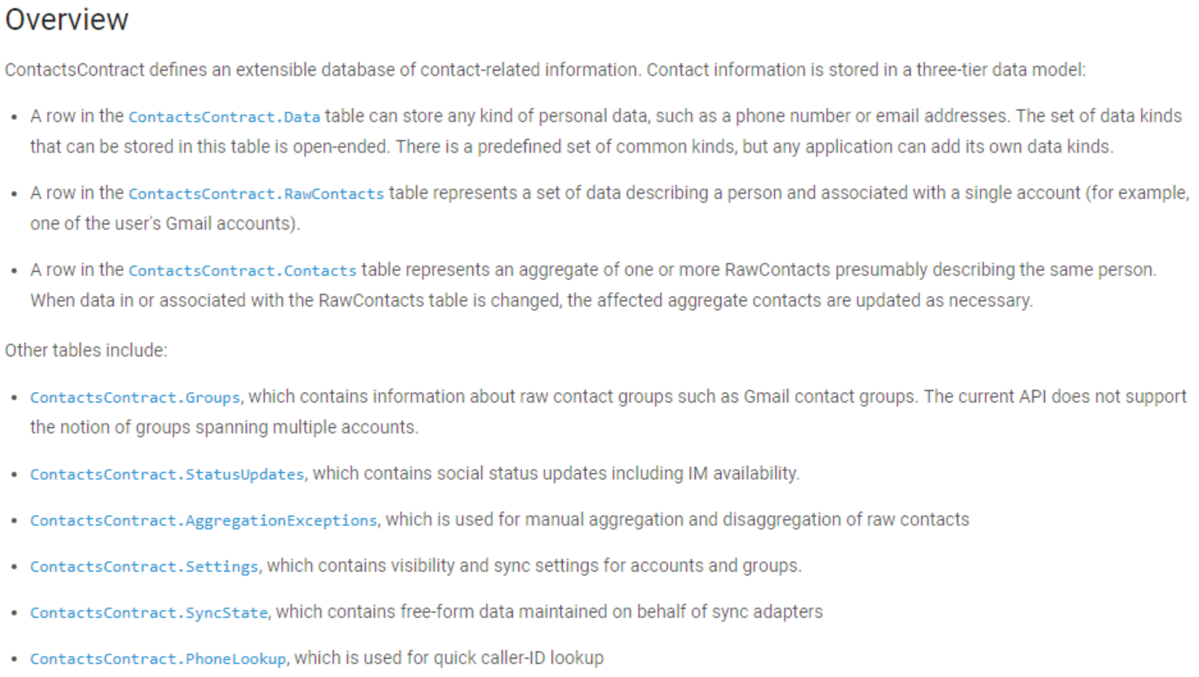
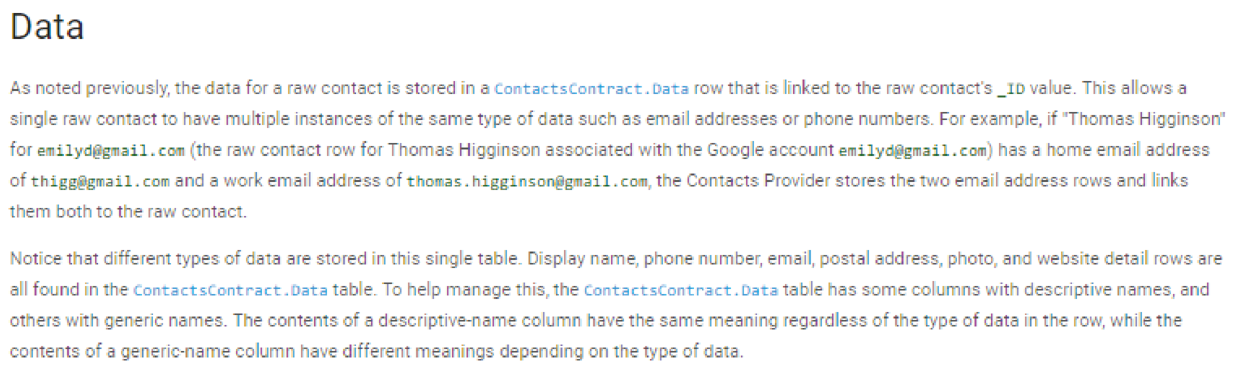
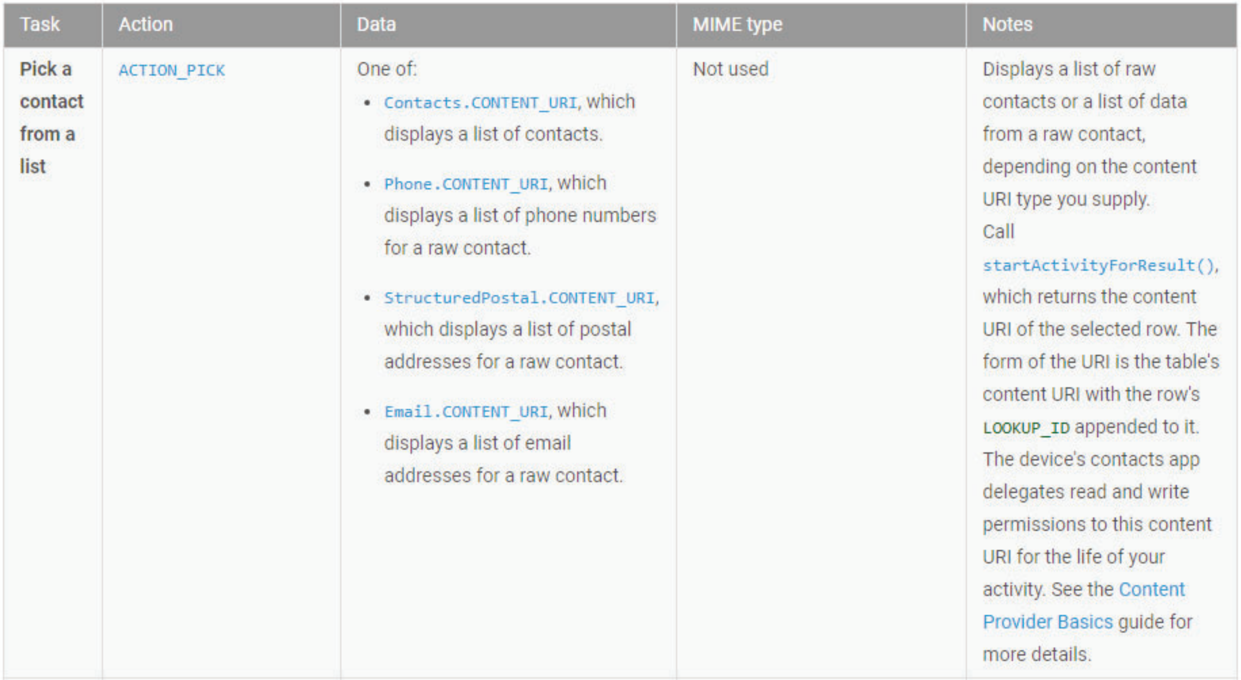
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Overview ContactsContract defines an extensible database of contact-related information. Contact information is stored in a three-tier data model: • A row in the `ContactsContract.Data` table can store any kind of personal data, such as a phone number or email addresses. The set of data kinds that can be stored in this table is open-ended. There is a predefined set of common kinds, but any application can add its own data kinds. • A row in the `ContactsContract.RawContacts` table represents a set of data describing a person and associated with a single account (for example, one of the user's Gmail accounts). • A row in the `ContactsContract.Contacts` table represents an aggregate of one or more RawContacts presumably describing the same person. When data in or associated with the RawContacts table is changed, the affected aggregate contacts are updated as necessary. Other tables include: • `ContactsContract.Groups`, which contains information about raw contact groups such as Gmail contact groups. The current API does not support the notion of groups spanning multiple accounts. • `ContactsContract.StatusUpdates`, which contains social status updates including IM availability. • `ContactsContract.AggregationExceptions`, which is used for manual aggregation and disaggregation of raw contacts • `ContactsContract.Settings`, which contains visibility and sync settings for accounts and groups. • `ContactsContract.SyncState`, which contains free-form data maintained on behalf of sync adapters • `ContactsContract.PhoneLookup`, which is used for quick caller-ID lookup https://developer.android.com/reference/android/provider/ContactsContract.html  Data As noted previously, the data for a raw contact is stored in a `ContactsContract.Data` row that is linked to the raw contact's `_ID` value. This allows a single raw contact to have multiple instances of the same type of data such as email addresses or phone numbers. For example, if "Thomas Higginson" for `emilyd@gmail.com` (the raw contact row for Thomas Higginson associated with the Google account `emilyd@gmail.com`) has a home email address of `thigg@gmail.com` and a work email address of `thomas.higginson@gmail.com`, the Contacts Provider stores the two email address rows and links them both to the raw contact. Notice that different types of data are stored in this single table. Display name, phone number, email, postal address, photo, and website detail rows are all found in the `ContactsContract.Data` table. To help manage this, the `ContactsContract.Data` table has some columns with descriptive names, and others with generic names. The contents of a descriptive-name column have the same meaning regardless of the type of data in the row, while the contents of a generic-name column have different meanings depending on the type of data.

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	https://developer.android.com/guide/topics/providers/contacts-provider.html

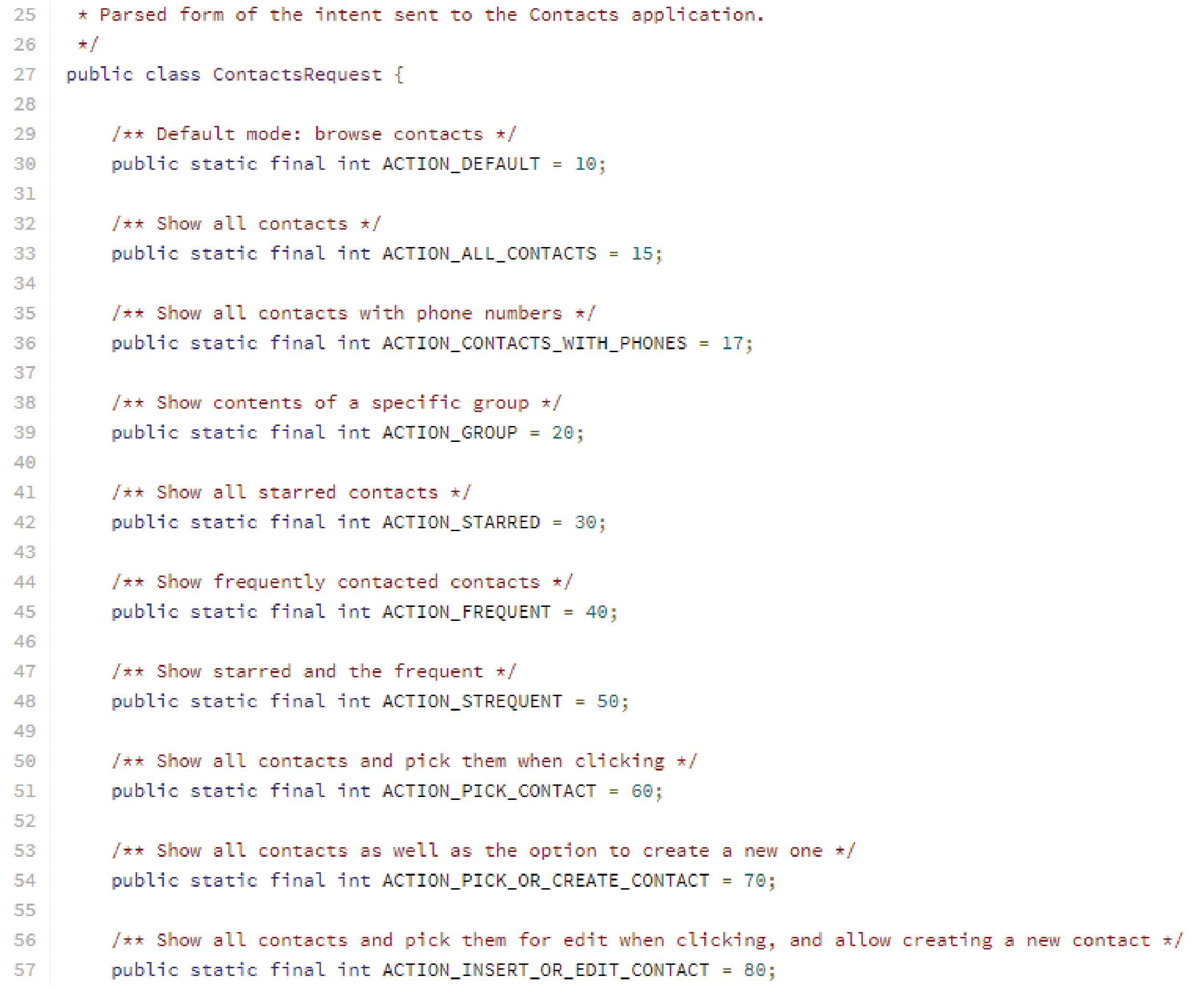


Task	Action	Data	MIME type	Notes
Pick a contact from a list	ACTION_PICK	One of: • Contacts.CONTENT_URI, which displays a list of contacts. • Phone.CONTENT_URI, which displays a list of phone numbers for a raw contact. • StructuredPostal.CONTENT_URI, which displays a list of postal addresses for a raw contact. • Email.CONTENT_URI, which displays a list of email addresses for a raw contact.	Not used	Displays a list of raw contacts or a list of data from a raw contact, depending on the content URI type you supply. Call startActivityForResult(), which returns the content URI of the selected row. The form of the URI is the table's content URI with the row's LOOKUP_ID appended to it. The device's contacts app delegates read and write permissions to this content URI for the life of your activity. See the Content Provider Basics guide for more details.

https://developer.android.com/guide/topics/providers/contacts-provider.html

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products

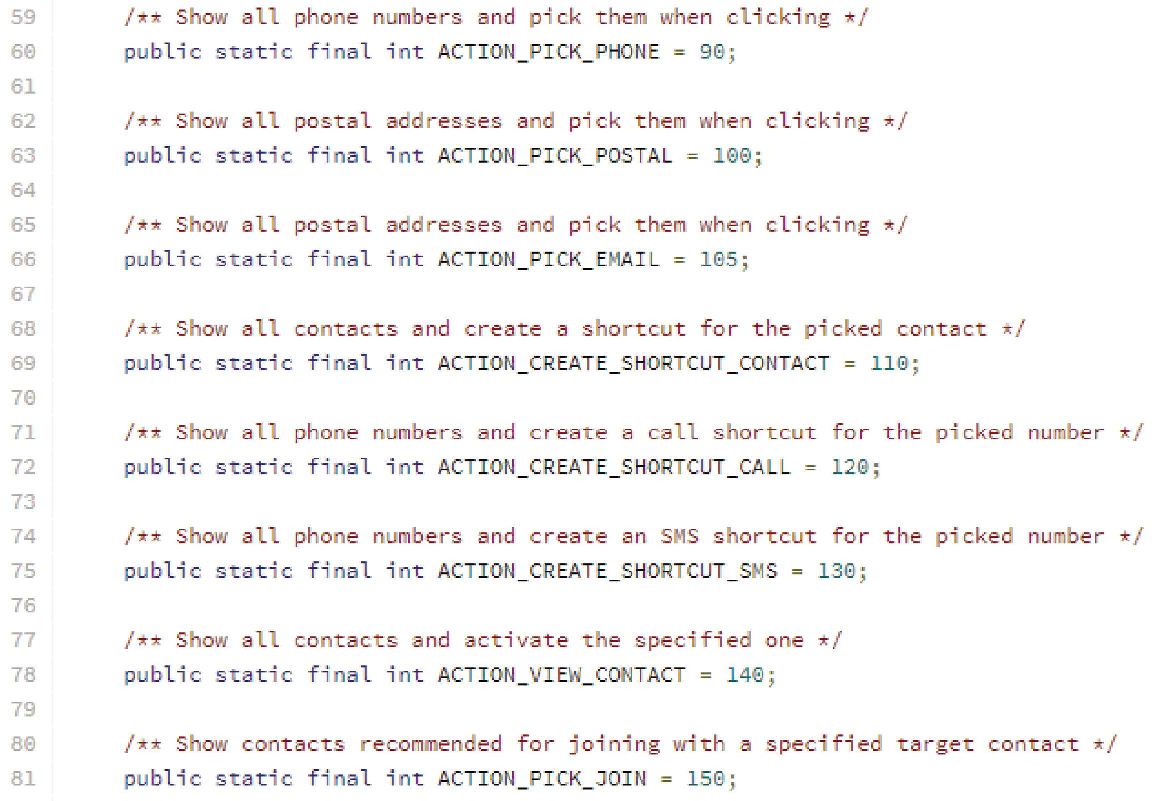


```
 * Parsed form of the intent sent to the Contacts application.
 */
public class ContactsRequest {

    /** Default mode: browse contacts */
    public static final int ACTION_DEFAULT = 10;

    /** Show all contacts */
    public static final int ACTION_ALL_CONTACTS = 15;

    /** Show all contacts with phone numbers */
    public static final int ACTION_CONTACTS_WITH_PHONES = 17;

    /** Show contents of a specific group */
    public static final int ACTION_GROUP = 20;

    /** Show all starred contacts */
    public static final int ACTION_STARRED = 30;

    /** Show frequently contacted contacts */
    public static final int ACTION_FREQUENT = 40;

    /** Show starred and the frequent */
    public static final int ACTION_STREQUENT = 50;

    /** Show all contacts and pick them when clicking */
    public static final int ACTION_PICK_CONTACT = 60;

    /** Show all contacts as well as the option to create a new one */
    public static final int ACTION_PICK_OR_CREATE_CONTACT = 70;

    /** Show all contacts and pick them for edit when clicking, and allow creating a new contact */
    public static final int ACTION_INSERT_OR_EDIT_CONTACT = 80;
```

https://android.googlesource.com/platform/packages/apps/Contacts/+/nougat-mr1-release/src/com/android/contacts/list/ContactsRequest.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

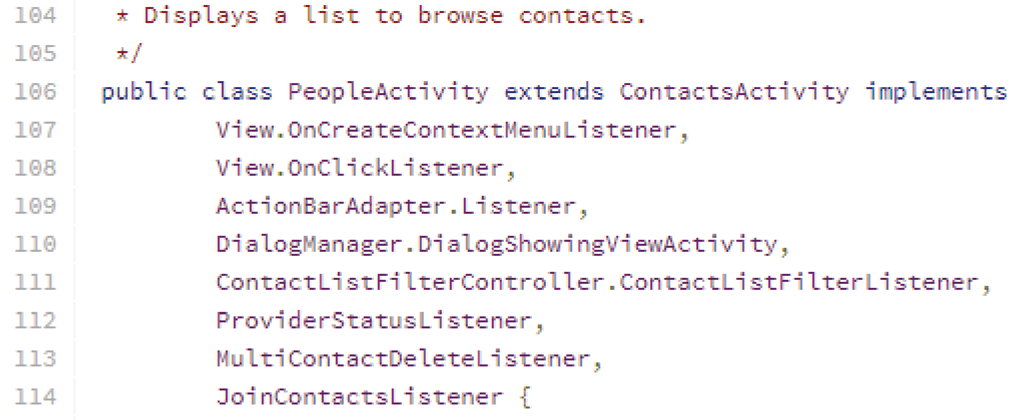
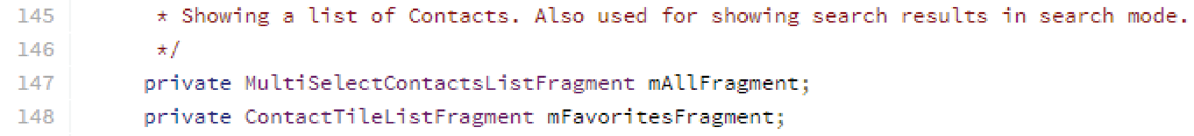
US9467838	Exemplary Supporting Evidence Regarding Accused Products



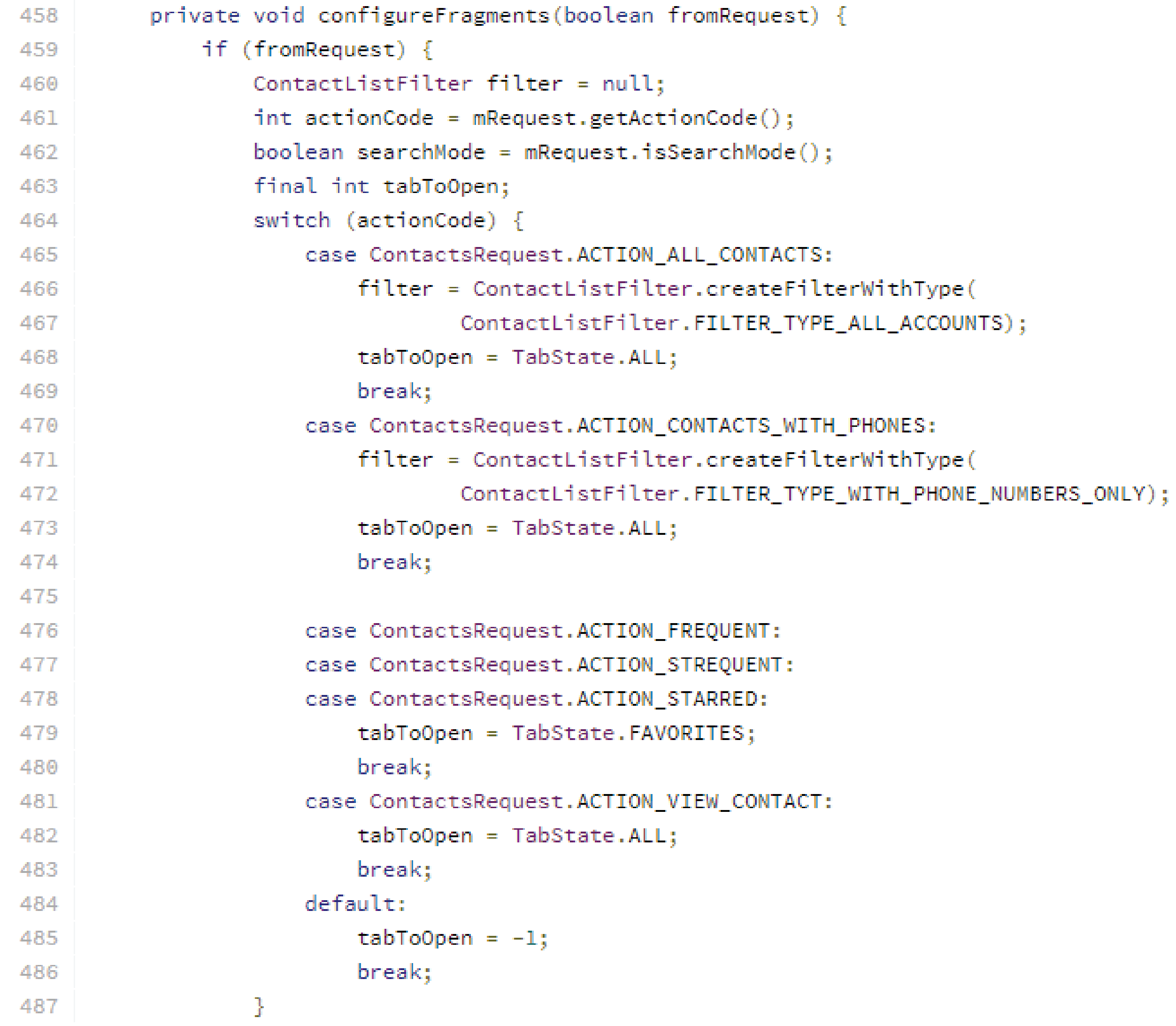
```
/** Show all phone numbers and pick them when clicking */
public static final int ACTION_PICK_PHONE = 90;

/** Show all postal addresses and pick them when clicking */
public static final int ACTION_PICK_POSTAL = 100;

/** Show all postal addresses and pick them when clicking */
public static final int ACTION_PICK_EMAIL = 105;

/** Show all contacts and create a shortcut for the picked contact */
public static final int ACTION_CREATE_SHORTCUT_CONTACT = 110;

/** Show all phone numbers and create a call shortcut for the picked number */
public static final int ACTION_CREATE_SHORTCUT_CALL = 120;

/** Show all phone numbers and create an SMS shortcut for the picked number */
public static final int ACTION_CREATE_SHORTCUT_SMS = 130;

/** Show all contacts and activate the specified one */
public static final int ACTION_VIEW_CONTACT = 140;

/** Show contacts recommended for joining with a specified target contact */
public static final int ACTION_PICK_JOIN = 150;
```

https://android.googlesource.com/platform/packages/apps/Contacts/+/nougat-mr1-release/src/com/android/contacts/list/ContactsRequest.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 * Displays a list to browse contacts. */ public class PeopleActivity extends ContactsActivity implements View.OnCreateContextMenuListener, View.OnClickListener, ActionBarAdapter.Listener, DialogManager.DialogShowingViewActivity, ContactListFilterController.ContactListFilterListener, ProviderStatusListener, MultiContactDeleteListener, JoinContactsListener { https://android.googlesource.com/platform/packages/apps/Contacts/+/nougat-mr1-release/src/com/android/contacts/activities/PeopleActivity.java  * Showing a list of Contacts. Also used for showing search results in search mode. */ private MultiSelectContactsListFragment mAllFragment; private ContactTileListFragment mFavoritesFragment; https://android.googlesource.com/platform/packages/apps/Contacts/+/nougat-mr1-release/src/com/android/contacts/activities/PeopleActivity.java

US9467838	Exemplary Supporting Evidence Regarding Accused Products



```
    private void configureFragments(boolean fromRequest) {
        if (fromRequest) {
            ContactListFilter filter = null;
            int actionCode = mRequest.getActionCode();
            boolean searchMode = mRequest.isSearchMode();
            final int tabToOpen;
            switch (actionCode) {
                case ContactsRequest.ACTION_ALL_CONTACTS:
                    filter = ContactListFilter.createFilterWithType(
                            ContactListFilter.FILTER_TYPE_ALL_ACCOUNTS);
                    tabToOpen = TabState.ALL;
                    break;
                case ContactsRequest.ACTION_CONTACTS_WITH_PHONES:
                    filter = ContactListFilter.createFilterWithType(
                            ContactListFilter.FILTER_TYPE_WITH_PHONE_NUMBERS_ONLY);
                    tabToOpen = TabState.ALL;
                    break;

                case ContactsRequest.ACTION_FREQUENT:
                case ContactsRequest.ACTION_STREQUENT:
                case ContactsRequest.ACTION_STARRED:
                    tabToOpen = TabState.FAVORITES;
                    break;
                case ContactsRequest.ACTION_VIEW_CONTACT:
                    tabToOpen = TabState.ALL;
                    break;
                default:
                    tabToOpen = -1;
                    break;
            }
```

https://android.googlesource.com/platform/packages/apps/Contacts/+/nougat-mr1-

US9467838	Exemplary Supporting Evidence Regarding Accused Products

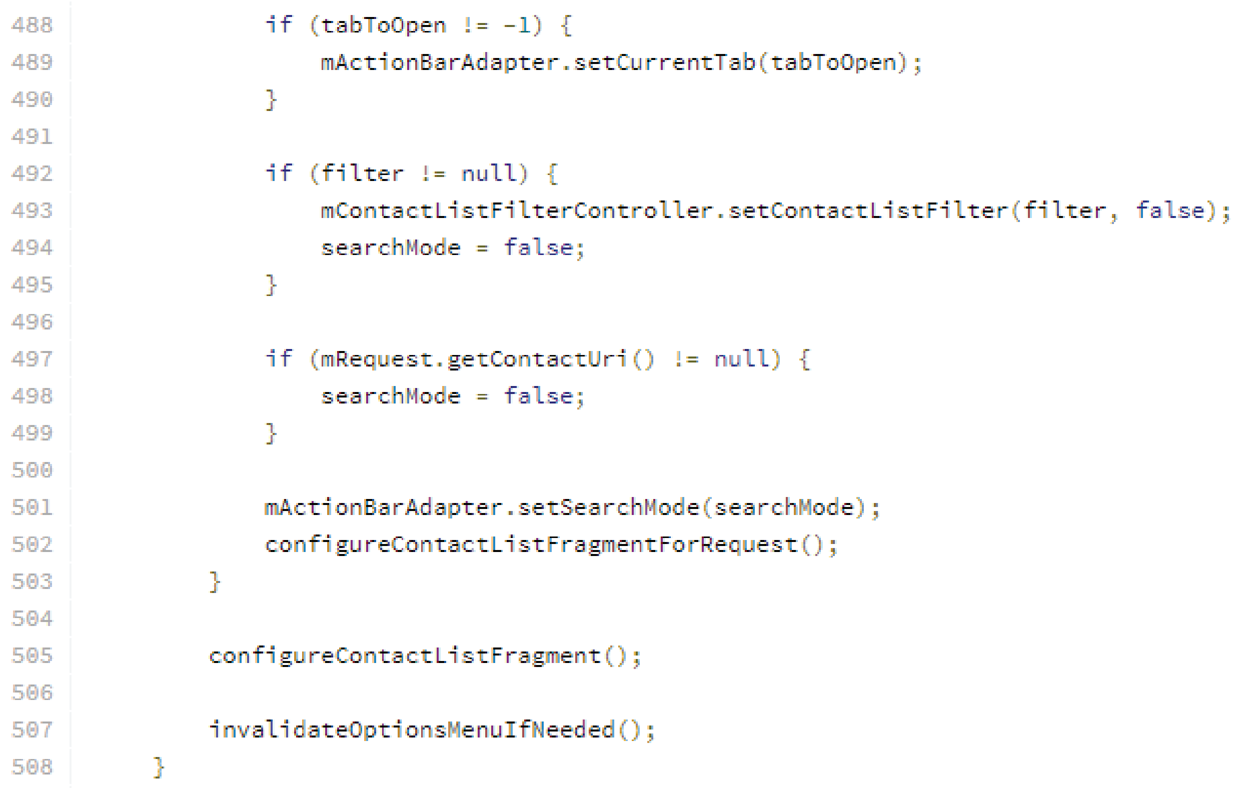
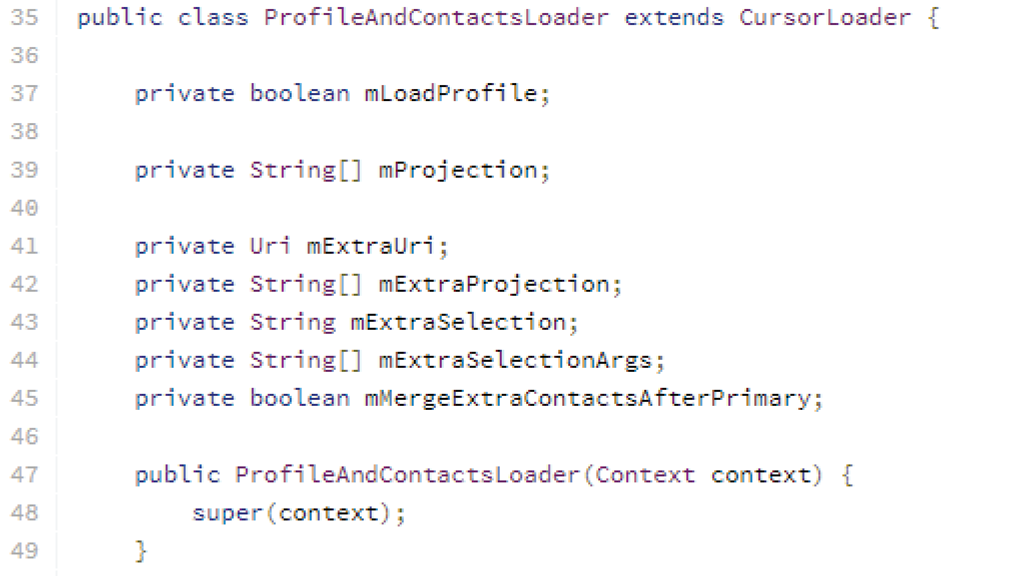
	release/src/com/android/contacts/activities/PeopleActivity.java
	
	`if (tabToOpen != -1) {`
	`    mActionBarAdapter.setCurrentTab(tabToOpen);`
	`}`
	
	`if (filter != null) {`
	`    mContactListFilterController.setContactListFilter(filter, false);`
	`    searchMode = false;`
	`}`
	
	`if (mRequest.getContactUri() != null) {`
	`    searchMode = false;`
	`}`
	
	`mActionBarAdapter.setSearchMode(searchMode);`
	`configureContactListFragmentForRequest();`
	`}`
	
	`configureContactListFragment();`
	
	`invalidateOptionsMenuIfNeeded();`
	`}`
	https://android.googlesource.com/platform/packages/apps/Contacts/+/nougat-mr1-release/src/com/android/contacts/activities/PeopleActivity.java

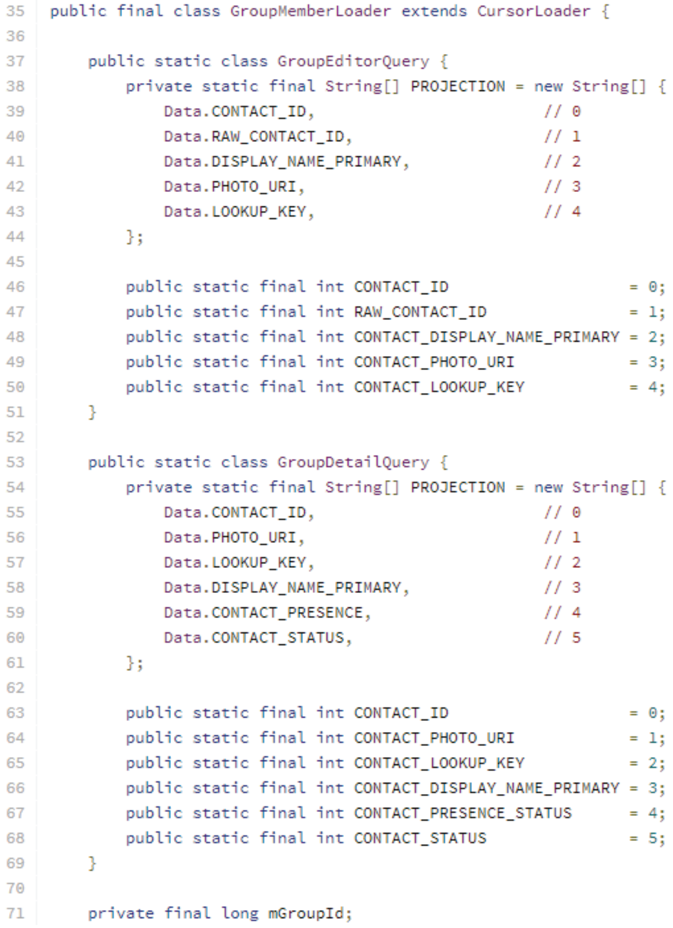
Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>


```
public class ProfileAndContactsLoader extends CursorLoader {

    private boolean mLoadProfile;

    private String[] mProjection;

    private Uri mExtraUri;
    private String[] mExtraProjection;
    private String mExtraSelection;
    private String[] mExtraSelectionArgs;
    private boolean mMergeExtraContactsAfterPrimary;

    public ProfileAndContactsLoader(Context context) {
        super(context);
    }
```

https://android.googlesource.com/platform/packages/apps/ContactsCommon/+/nougat-mr1-release/src/com/android/contacts/common/list/ProfileAndContactsLoader.java
</td></tr>
</table>

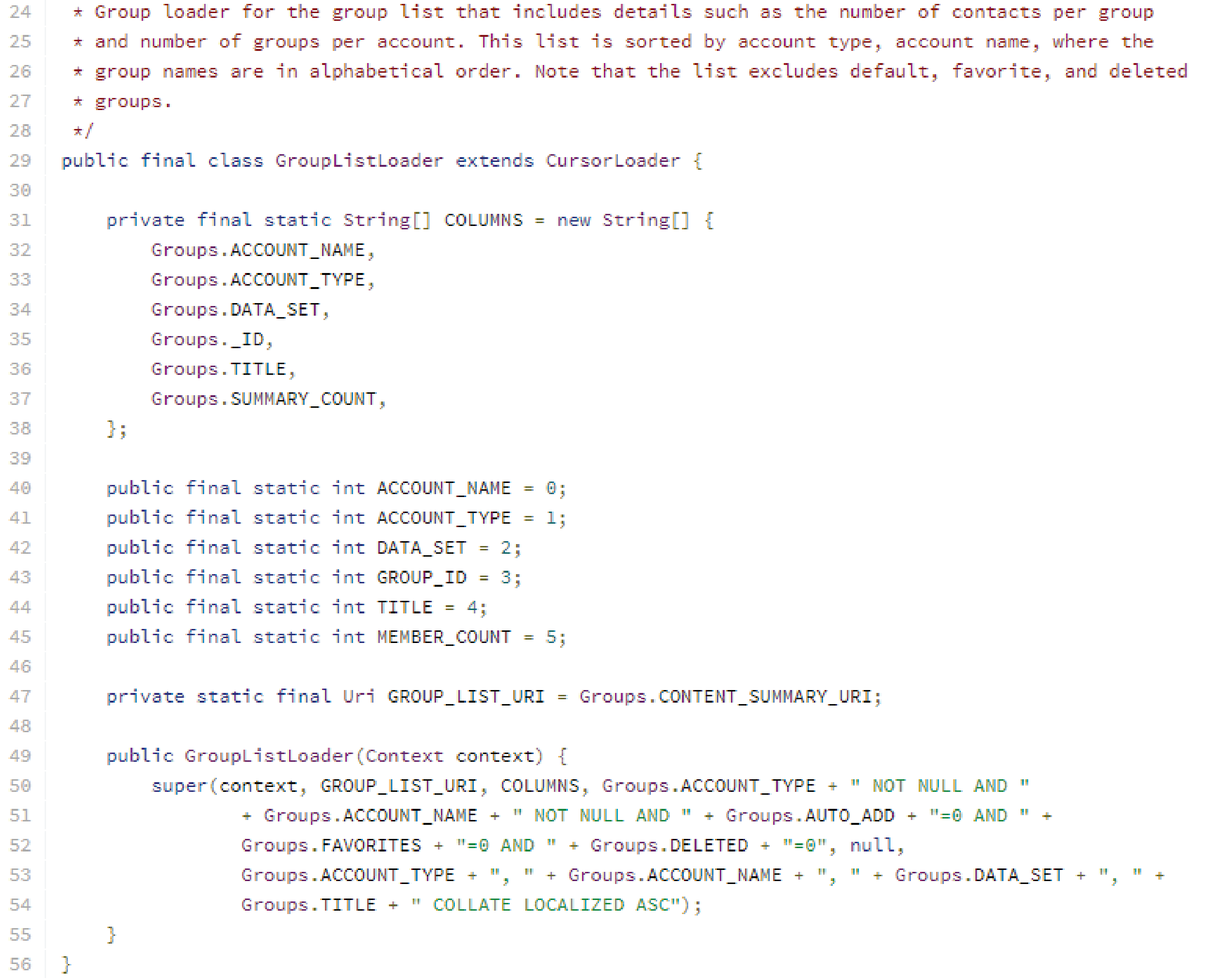
Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>


```
public final class GroupMemberLoader extends CursorLoader {

    public static class GroupEditorQuery {
        private static final String[] PROJECTION = new String[] {
            Data.CONTACT_ID,                        // 0
            Data.RAW_CONTACT_ID,                    // 1
            Data.DISPLAY_NAME_PRIMARY,              // 2
            Data.PHOTO_URI,                         // 3
            Data.LOOKUP_KEY,                        // 4
        };

        public static final int CONTACT_ID                   = 0;
        public static final int RAW_CONTACT_ID               = 1;
        public static final int CONTACT_DISPLAY_NAME_PRIMARY = 2;
        public static final int CONTACT_PHOTO_URI            = 3;
        public static final int CONTACT_LOOKUP_KEY           = 4;
    }

    public static class GroupDetailQuery {
        private static final String[] PROJECTION = new String[] {
            Data.CONTACT_ID,                        // 0
            Data.PHOTO_URI,                         // 1
            Data.LOOKUP_KEY,                        // 2
            Data.DISPLAY_NAME_PRIMARY,              // 3
            Data.CONTACT_PRESENCE,                  // 4
            Data.CONTACT_STATUS,                    // 5
        };

        public static final int CONTACT_ID                   = 0;
        public static final int CONTACT_PHOTO_URI            = 1;
        public static final int CONTACT_LOOKUP_KEY           = 2;
        public static final int CONTACT_DISPLAY_NAME_PRIMARY = 3;
        public static final int CONTACT_PRESENCE_STATUS      = 4;
        public static final int CONTACT_STATUS               = 5;
    }

    private final long mGroupId;
```

https://android.googlesource.com/platform/packages/apps/Contacts/+/nougat-mr1-release/src/com/android/contacts/GroupMemberLoader.java
</td></tr>
</table>

US9467838	Exemplary Supporting Evidence Regarding Accused Products

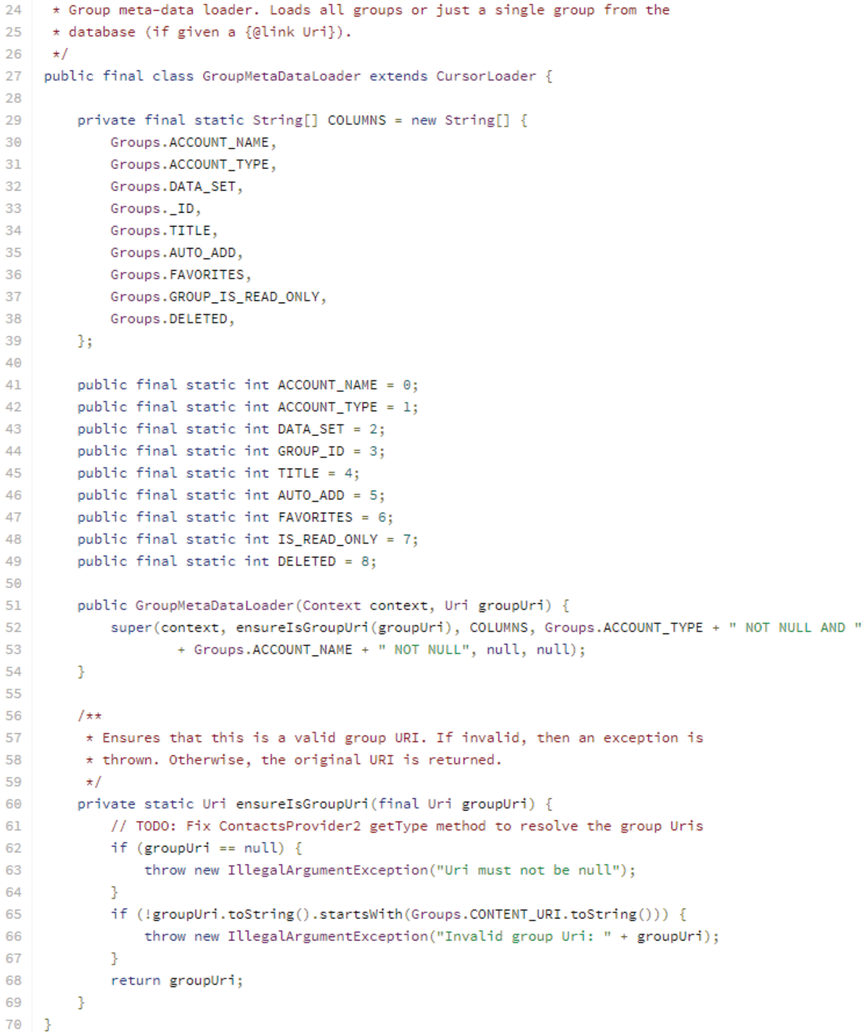


```
 * Group loader for the group list that includes details such as the number of contacts per group
 * and number of groups per account. This list is sorted by account type, account name, where the
 * group names are in alphabetical order. Note that the list excludes default, favorite, and deleted
 * groups.
 */
public final class GroupListLoader extends CursorLoader {

    private final static String[] COLUMNS = new String[] {
        Groups.ACCOUNT_NAME,
        Groups.ACCOUNT_TYPE,
        Groups.DATA_SET,
        Groups._ID,
        Groups.TITLE,
        Groups.SUMMARY_COUNT,
    };

    public final static int ACCOUNT_NAME = 0;
    public final static int ACCOUNT_TYPE = 1;
    public final static int DATA_SET = 2;
    public final static int GROUP_ID = 3;
    public final static int TITLE = 4;
    public final static int MEMBER_COUNT = 5;

    private static final Uri GROUP_LIST_URI = Groups.CONTENT_SUMMARY_URI;

    public GroupListLoader(Context context) {
        super(context, GROUP_LIST_URI, COLUMNS, Groups.ACCOUNT_TYPE + " NOT NULL AND "
                + Groups.ACCOUNT_NAME + " NOT NULL AND " + Groups.AUTO_ADD + "=0 AND " +
                Groups.FAVORITES + "=0 AND " + Groups.DELETED + "=0", null,
                Groups.ACCOUNT_TYPE + ", " + Groups.ACCOUNT_NAME + ", " + Groups.DATA_SET + ", " +
                Groups.TITLE + " COLLATE LOCALIZED ASC");
    }
}
```

https://android.googlesource.com/platform/packages/apps/Contacts/+/nougat-mr1-release/src/com/android/contacts/GroupListLoader.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

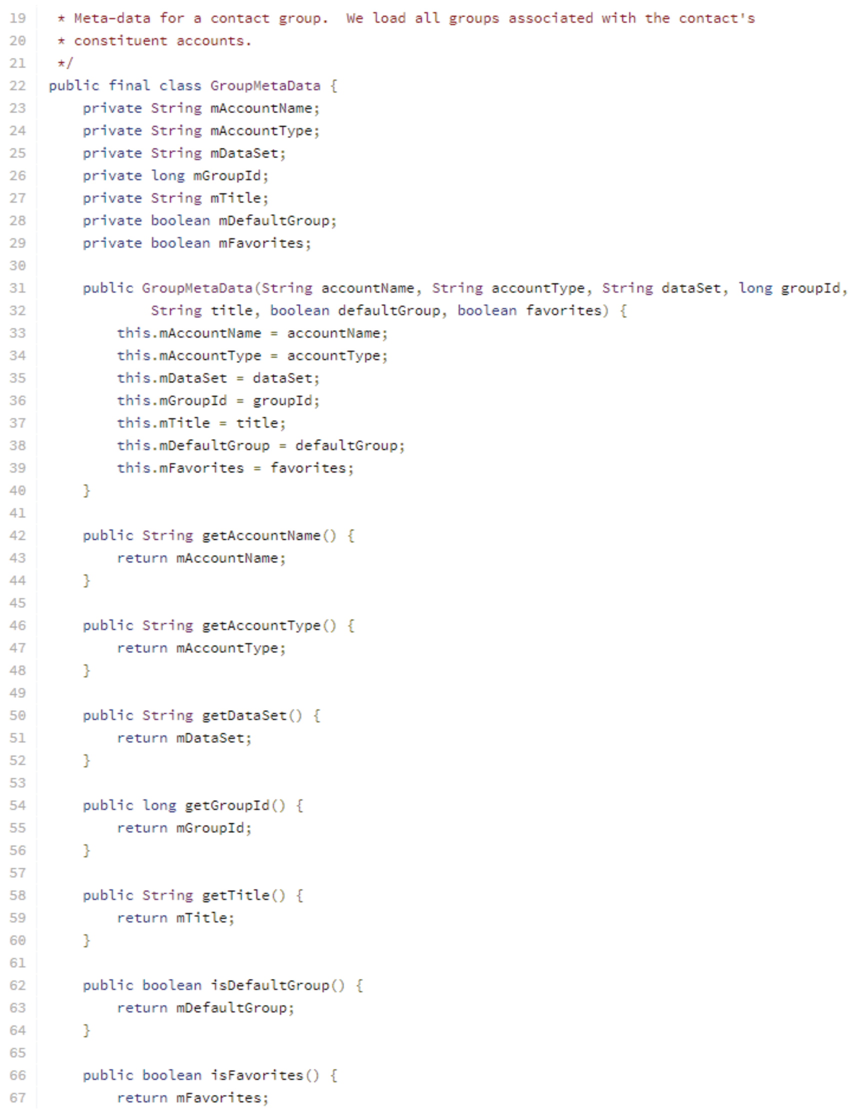
US9467838	Exemplary Supporting Evidence Regarding Accused Products



```
 * Group meta-data loader. Loads all groups or just a single group from the
 * database (if given a {@link Uri}).
 */
public final class GroupMetaDataLoader extends CursorLoader {

    private final static String[] COLUMNS = new String[] {
        Groups.ACCOUNT_NAME,
        Groups.ACCOUNT_TYPE,
        Groups.DATA_SET,
        Groups._ID,
        Groups.TITLE,
        Groups.AUTO_ADD,
        Groups.FAVORITES,
        Groups.GROUP_IS_READ_ONLY,
        Groups.DELETED,
    };

    public final static int ACCOUNT_NAME = 0;
    public final static int ACCOUNT_TYPE = 1;
    public final static int DATA_SET = 2;
    public final static int GROUP_ID = 3;
    public final static int TITLE = 4;
    public final static int AUTO_ADD = 5;
    public final static int FAVORITES = 6;
    public final static int IS_READ_ONLY = 7;
    public final static int DELETED = 8;

    public GroupMetaDataLoader(Context context, Uri groupUri) {
        super(context, ensureIsGroupUri(groupUri), COLUMNS, Groups.ACCOUNT_TYPE + " NOT NULL AND "
                + Groups.ACCOUNT_NAME + " NOT NULL", null, null);
    }

    /**
     * Ensures that this is a valid group URI. If invalid, then an exception is
     * thrown. Otherwise, the original URI is returned.
     */
    private static Uri ensureIsGroupUri(final Uri groupUri) {
        // TODO: Fix ContactsProvider2 getType method to resolve the group Uris
        if (groupUri == null) {
            throw new IllegalArgumentException("Uri must not be null");
        }
        if (!groupUri.toString().startsWith(Groups.CONTENT_URI.toString())) {
            throw new IllegalArgumentException("Invalid group Uri: " + groupUri);
        }
        return groupUri;
    }
}
```

https://android.googlesource.com/platform/packages/apps/Contacts/+/nougat-mr1-release/src/com/android/contacts/GroupMetaDataLoader.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

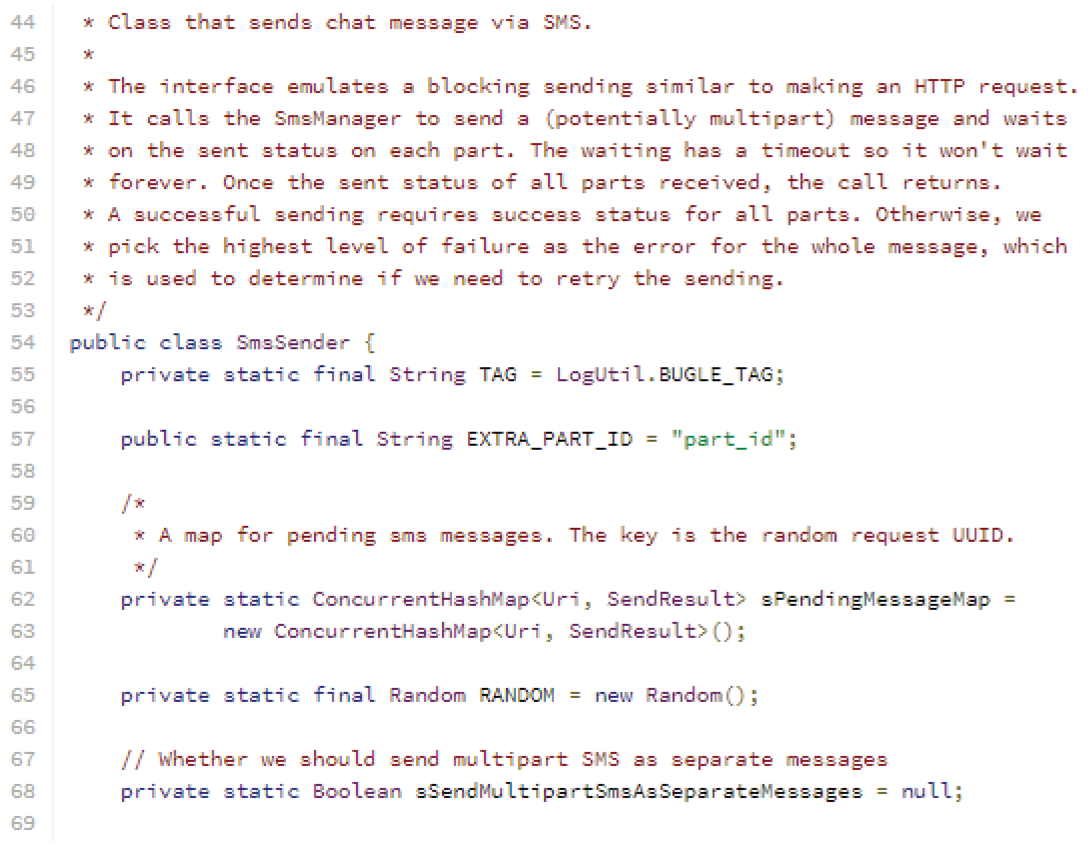
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
 * Meta-data for a contact group.  We load all groups associated with the contact's
 * constituent accounts.
 */
public final class GroupMetaData {
    private String mAccountName;
    private String mAccountType;
    private String mDataSet;
    private long mGroupId;
    private String mTitle;
    private boolean mDefaultGroup;
    private boolean mFavorites;

    public GroupMetaData(String accountName, String accountType, String dataSet, long groupId,
            String title, boolean defaultGroup, boolean favorites) {
        this.mAccountName = accountName;
        this.mAccountType = accountType;
        this.mDataSet = dataSet;
        this.mGroupId = groupId;
        this.mTitle = title;
        this.mDefaultGroup = defaultGroup;
        this.mFavorites = favorites;
    }

    public String getAccountName() {
        return mAccountName;
    }

    public String getAccountType() {
        return mAccountType;
    }

    public String getDataSet() {
        return mDataSet;
    }

    public long getGroupId() {
        return mGroupId;
    }

    public String getTitle() {
        return mTitle;
    }

    public boolean isDefaultGroup() {
        return mDefaultGroup;
    }

    public boolean isFavorites() {
        return mFavorites;
```

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>

https://android.googlesource.com/platform/packages/apps/ContactsCommon/+/nougat-mr1-release/src/com/android/contacts/common/GroupMetaData.java

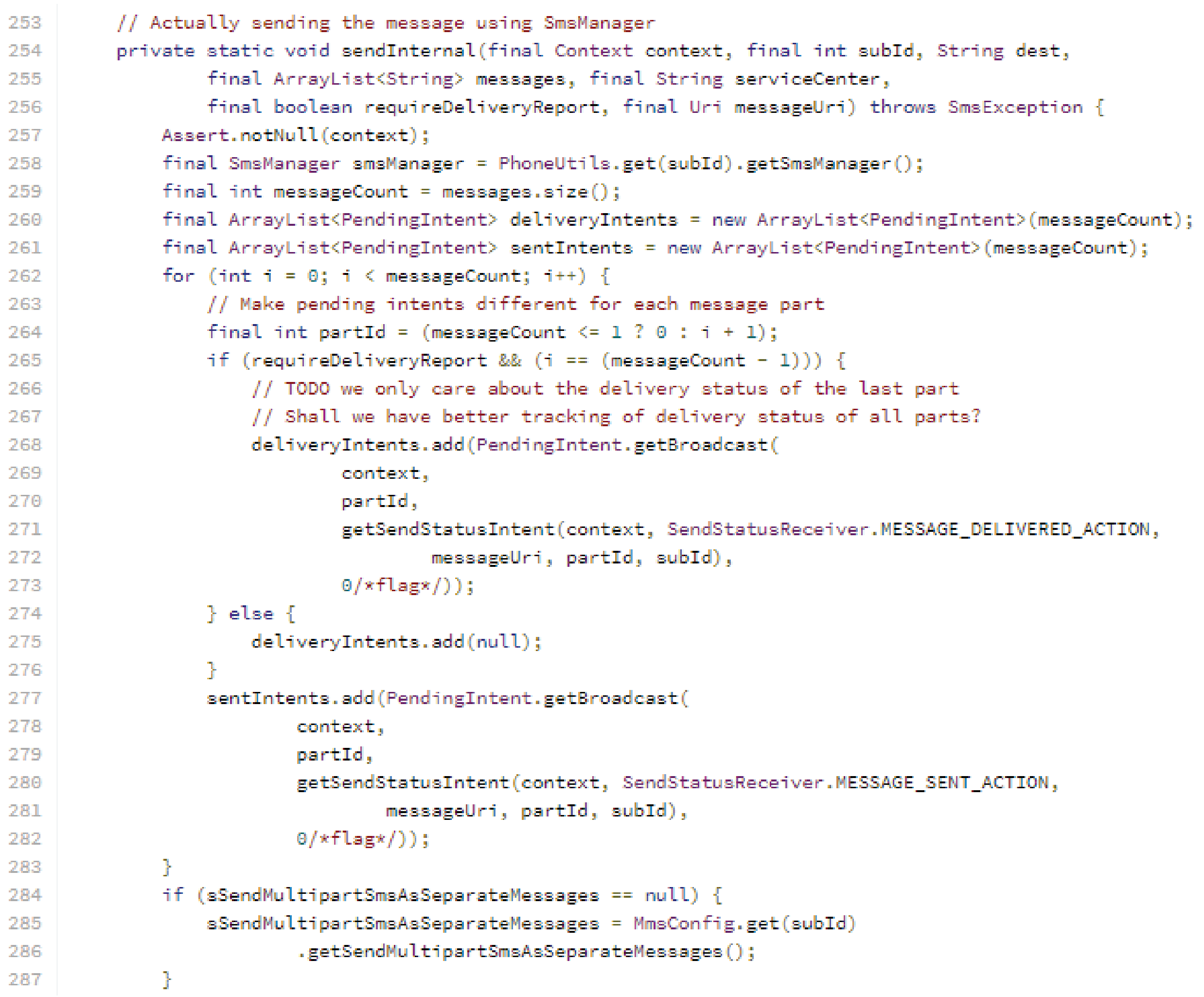


```
 * Class that sends chat message via SMS.
 *
 * The interface emulates a blocking sending similar to making an HTTP request.
 * It calls the SmsManager to send a (potentially multipart) message and waits
 * on the sent status on each part. The waiting has a timeout so it won't wait
 * forever. Once the sent status of all parts received, the call returns.
 * A successful sending requires success status for all parts. Otherwise, we
 * pick the highest level of failure as the error for the whole message, which
 * is used to determine if we need to retry the sending.
 */
public class SmsSender {
    private static final String TAG = LogUtil.BUGLE_TAG;

    public static final String EXTRA_PART_ID = "part_id";

    /*
     * A map for pending sms messages. The key is the random request UUID.
     */
    private static ConcurrentHashMap<Uri, SendResult> sPendingMessageMap =
            new ConcurrentHashMap<Uri, SendResult>();

    private static final Random RANDOM = new Random();

    // Whether we should send multipart SMS as separate messages
    private static Boolean sSendMultipartSmsAsSeparateMessages = null;

```

</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/SmsSender.java



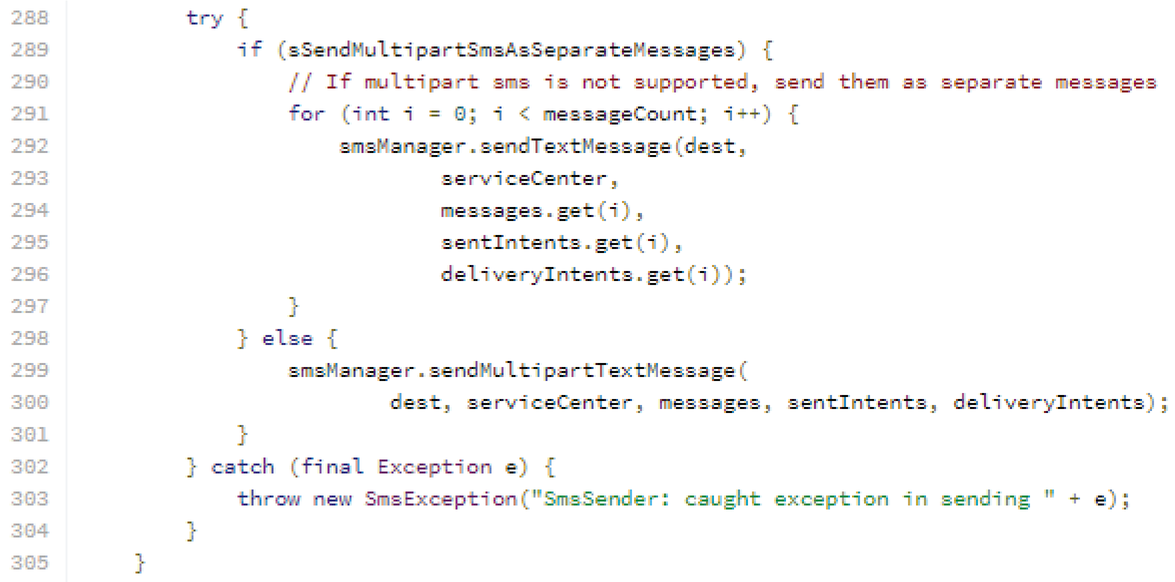
```
    // Actually sending the message using SmsManager
    private static void sendInternal(final Context context, final int subId, String dest,
            final ArrayList<String> messages, final String serviceCenter,
            final boolean requireDeliveryReport, final Uri messageUri) throws SmsException {
        Assert.notNull(context);
        final SmsManager smsManager = PhoneUtils.get(subId).getSmsManager();
        final int messageCount = messages.size();
        final ArrayList<PendingIntent> deliveryIntents = new ArrayList<PendingIntent>(messageCount);
        final ArrayList<PendingIntent> sentIntents = new ArrayList<PendingIntent>(messageCount);
        for (int i = 0; i < messageCount; i++) {
            // Make pending intents different for each message part
            final int partId = (messageCount <= 1 ? 0 : i + 1);
            if (requireDeliveryReport && (i == (messageCount - 1))) {
                // TODO we only care about the delivery status of the last part
                // Shall we have better tracking of delivery status of all parts?
                deliveryIntents.add(PendingIntent.getBroadcast(
                        context,
                        partId,
                        getSendStatusIntent(context, SendStatusReceiver.MESSAGE_DELIVERED_ACTION,
                                messageUri, partId, subId),
                        0/*flag*/));
            } else {
                deliveryIntents.add(null);
            }
            sentIntents.add(PendingIntent.getBroadcast(
                    context,
                    partId,
                    getSendStatusIntent(context, SendStatusReceiver.MESSAGE_SENT_ACTION,
                            messageUri, partId, subId),
                    0/*flag*/));
        }
        if (sSendMultipartSmsAsSeparateMessages == null) {
            sSendMultipartSmsAsSeparateMessages = MmsConfig.get(subId)
                    .getSendMultipartSmsAsSeparateMessages();
        }
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>

release/src/com/android/messaging/sms/SmsSender.java

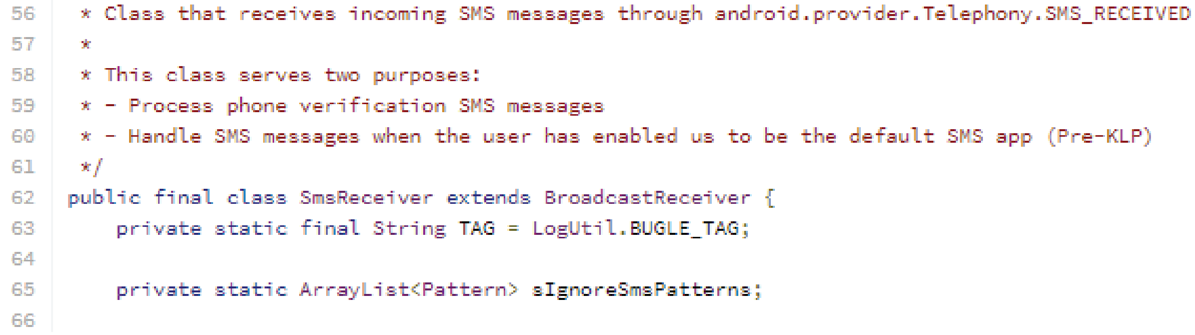


```
        try {
            if (sSendMultipartSmsAsSeparateMessages) {
                // If multipart sms is not supported, send them as separate messages
                for (int i = 0; i < messageCount; i++) {
                    smsManager.sendTextMessage(dest,
                            serviceCenter,
                            messages.get(i),
                            sentIntents.get(i),
                            deliveryIntents.get(i));
                }
            } else {
                smsManager.sendMultipartTextMessage(
                        dest, serviceCenter, messages, sentIntents, deliveryIntents);
            }
        } catch (final Exception e) {
            throw new SmsException("SmsSender: caught exception in sending " + e);
        }
    }
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/SmsSender.java

</td></tr>
</table>

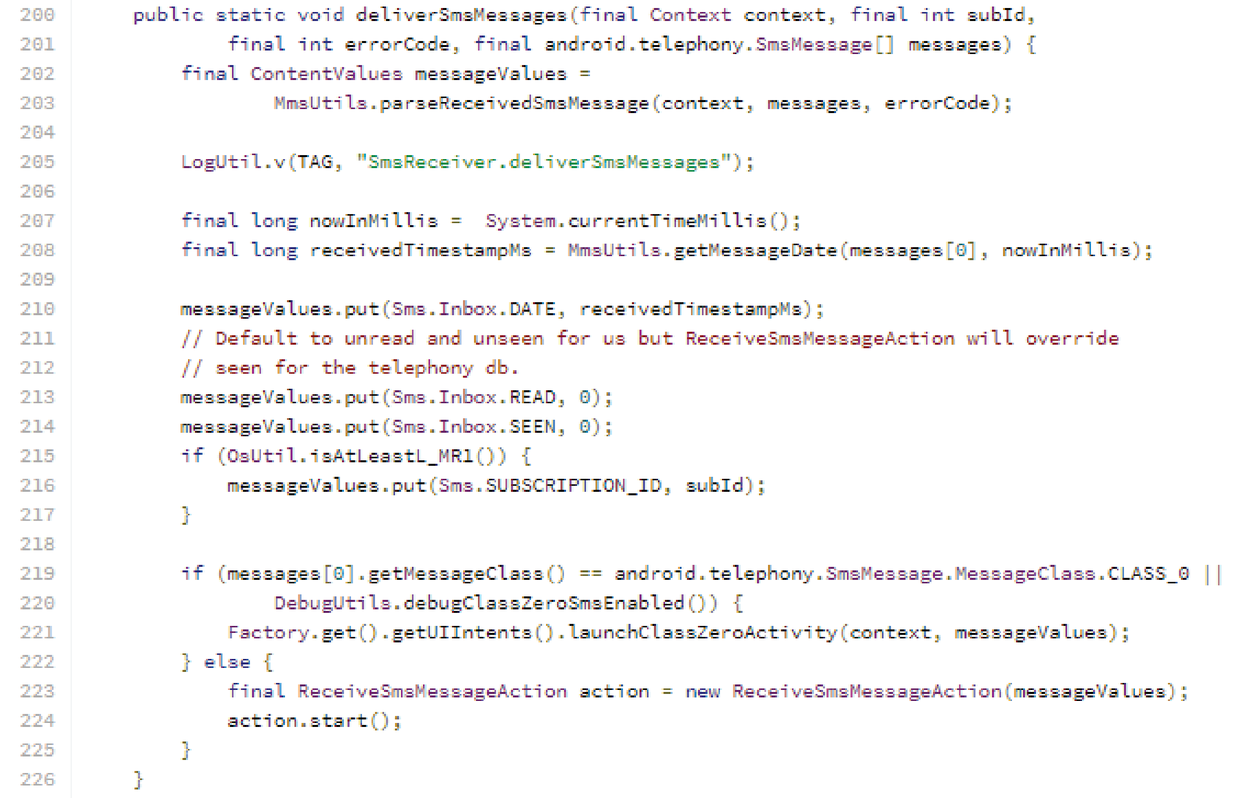
Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
 * Class that receives incoming SMS messages through android.provider.Telephony.SMS_RECEIVED
 *
 * This class serves two purposes:
 * - Process phone verification SMS messages
 * - Handle SMS messages when the user has enabled us to be the default SMS app (Pre-KLP)
 */
public final class SmsReceiver extends BroadcastReceiver {
    private static final String TAG = LogUtil.BUGLE_TAG;

    private static ArrayList<Pattern> sIgnoreSmsPatterns;

```

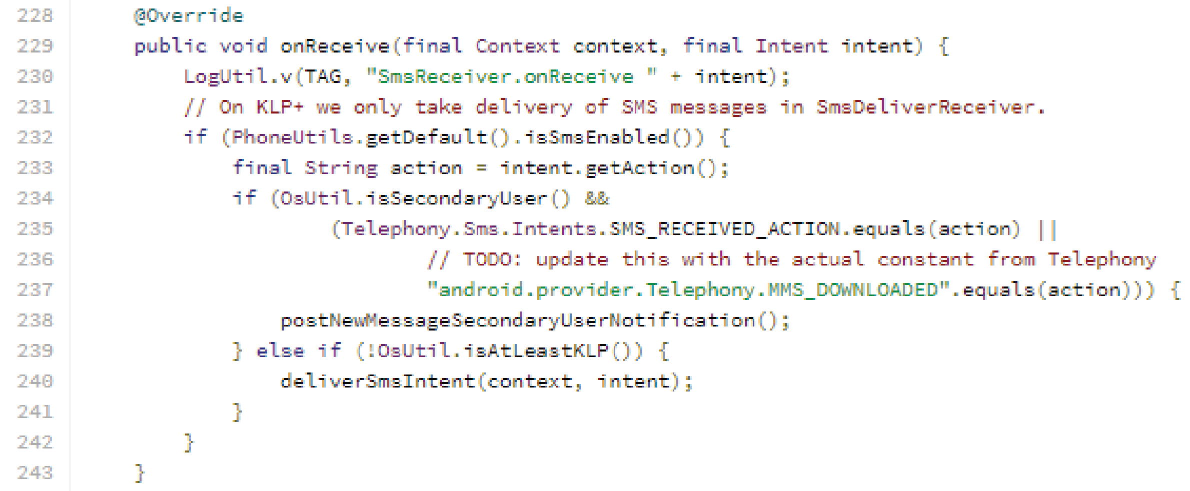
https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/receiver/SmsReceiver.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
    public static void deliverSmsMessages(final Context context, final int subId,
            final int errorCode, final android.telephony.SmsMessage[] messages) {
        final ContentValues messageValues =
                MmsUtils.parseReceivedSmsMessage(context, messages, errorCode);

        LogUtil.v(TAG, "SmsReceiver.deliverSmsMessages");

        final long nowInMillis =  System.currentTimeMillis();
        final long receivedTimestampMs = MmsUtils.getMessageDate(messages[0], nowInMillis);

        messageValues.put(Sms.Inbox.DATE, receivedTimestampMs);
        // Default to unread and unseen for us but ReceiveSmsMessageAction will override
        // seen for the telephony db.
        messageValues.put(Sms.Inbox.READ, 0);
        messageValues.put(Sms.Inbox.SEEN, 0);
        if (OsUtil.isAtLeastL_MR1()) {
            messageValues.put(Sms.SUBSCRIPTION_ID, subId);
        }

        if (messages[0].getMessageClass() == android.telephony.SmsMessage.MessageClass.CLASS_0 ||
                DebugUtils.debugClassZeroSmsEnabled()) {
            Factory.get().getUIIntents().launchClassZeroActivity(context, messageValues);
        } else {
            final ReceiveSmsMessageAction action = new ReceiveSmsMessageAction(messageValues);
            action.start();
        }
    }
```

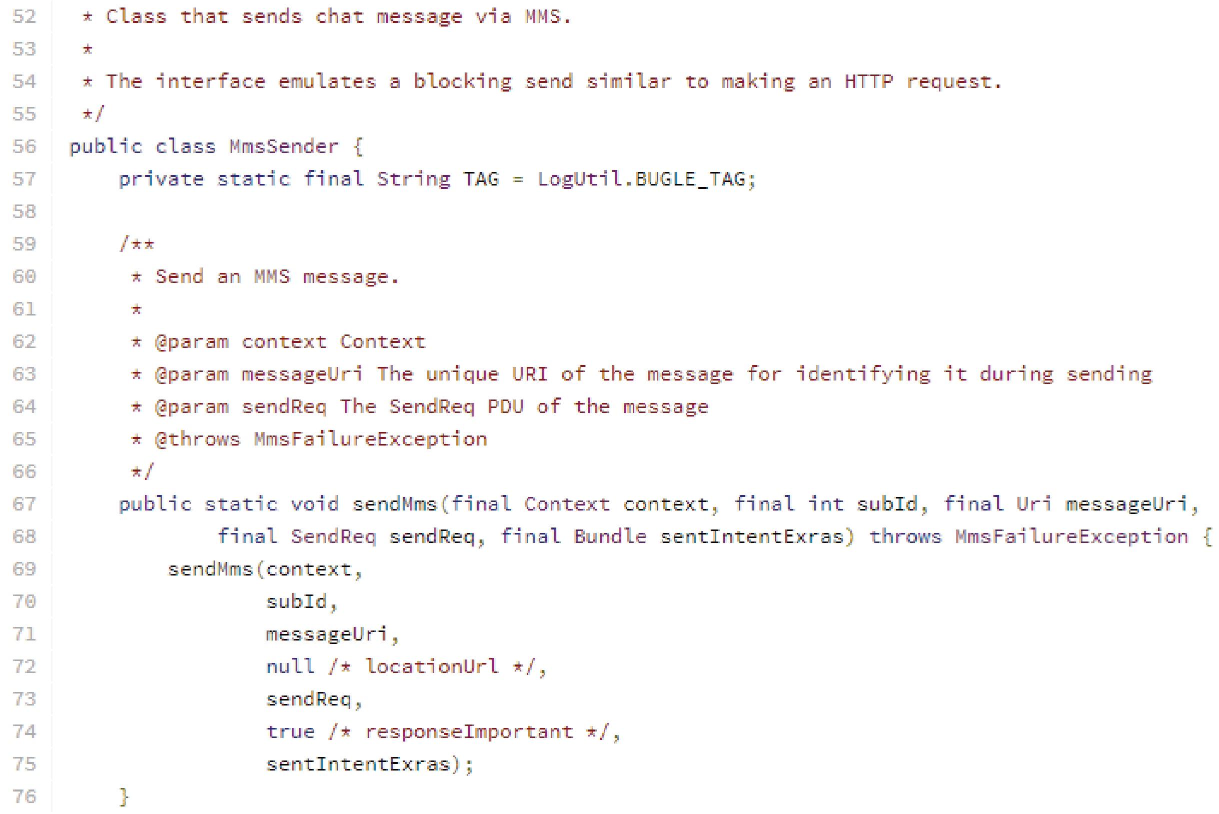
https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/receiver/SmsReceiver.java

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/receiver/SmsReceiver.java

```
    @Override
    public void onReceive(final Context context, final Intent intent) {
        LogUtil.v(TAG, "SmsReceiver.onReceive " + intent);
        // On KLP+ we only take delivery of SMS messages in SmsDeliverReceiver.
        if (PhoneUtils.getDefault().isSmsEnabled()) {
            final String action = intent.getAction();
            if (OsUtil.isSecondaryUser() &&
                    (Telephony.Sms.Intents.SMS_RECEIVED_ACTION.equals(action) ||
                            // TODO: update this with the actual constant from Telephony
                            "android.provider.Telephony.MMS_DOWNLOADED".equals(action))) {
                postNewMessageSecondaryUserNotification();
            } else if (!OsUtil.isAtLeastKLP()) {
                deliverSmsIntent(context, intent);
            }
        }
    }
```

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products

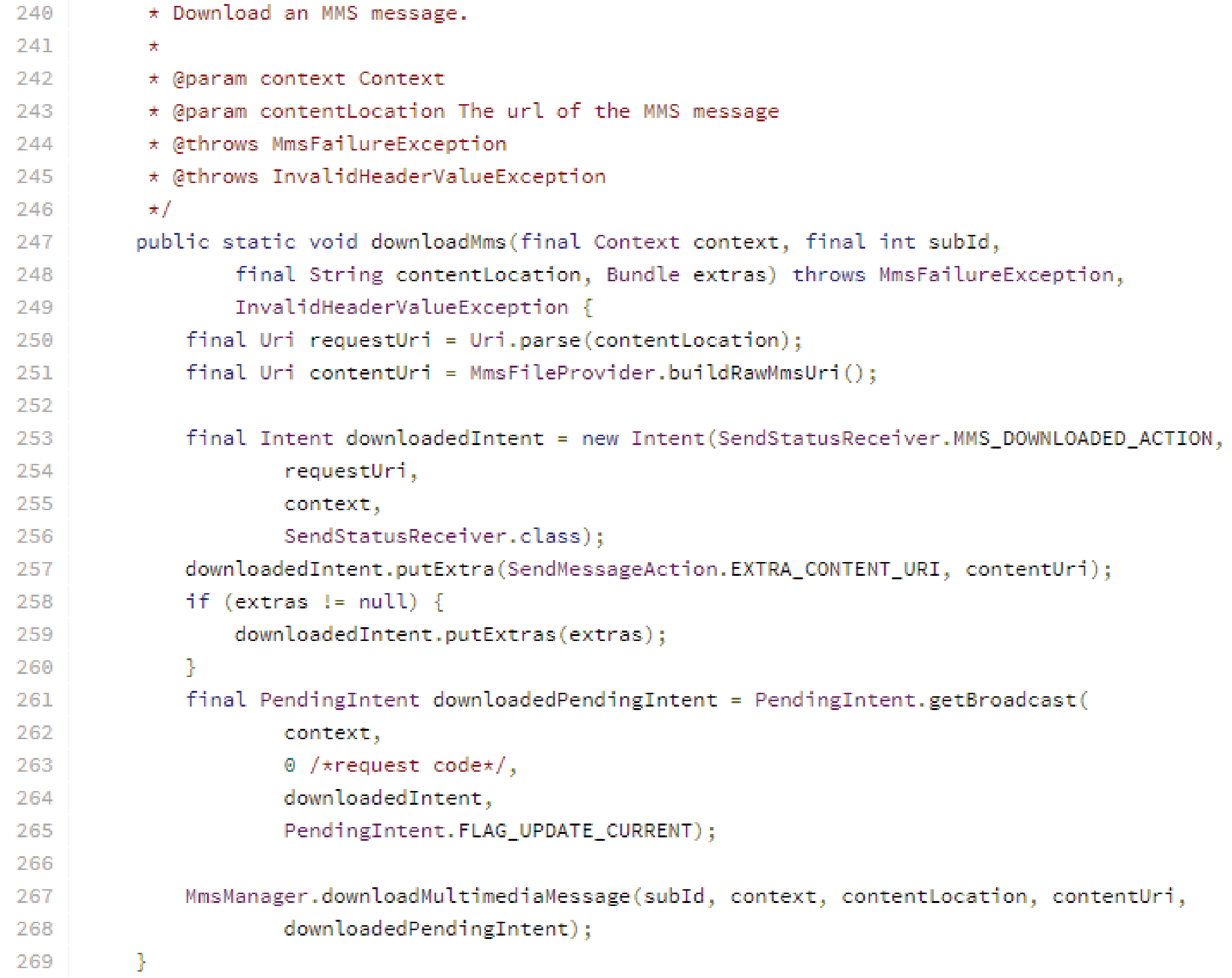


```
 * Class that sends chat message via MMS.
 *
 * The interface emulates a blocking send similar to making an HTTP request.
 */
public class MmsSender {
    private static final String TAG = LogUtil.BUGLE_TAG;

    /**
     * Send an MMS message.
     *
     * @param context Context
     * @param messageUri The unique URI of the message for identifying it during sending
     * @param sendReq The SendReq PDU of the message
     * @throws MmsFailureException
     */
    public static void sendMms(final Context context, final int subId, final Uri messageUri,
            final SendReq sendReq, final Bundle sentIntentExras) throws MmsFailureException {
        sendMms(context,
                subId,
                messageUri,
                null /* locationUrl */,
                sendReq,
                true /* responseImportant */,
                sentIntentExras);
    }
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/MmsSender.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>



```
 * Download an MMS message.
 *
 * @param context Context
 * @param contentLocation The url of the MMS message
 * @throws MmsFailureException
 * @throws InvalidHeaderValueException
 */
public static void downloadMms(final Context context, final int subId,
        final String contentLocation, Bundle extras) throws MmsFailureException,
        InvalidHeaderValueException {
    final Uri requestUri = Uri.parse(contentLocation);
    final Uri contentUri = MmsFileProvider.buildRawMmsUri();

    final Intent downloadedIntent = new Intent(SendStatusReceiver.MMS_DOWNLOADED_ACTION,
            requestUri,
            context,
            SendStatusReceiver.class);
    downloadedIntent.putExtra(SendMessageAction.EXTRA_CONTENT_URI, contentUri);
    if (extras != null) {
        downloadedIntent.putExtras(extras);
    }
    final PendingIntent downloadedPendingIntent = PendingIntent.getBroadcast(
            context,
            0 /*request code*/,
            downloadedIntent,
            PendingIntent.FLAG_UPDATE_CURRENT);

    MmsManager.downloadMultimediaMessage(subId, context, contentLocation, contentUri,
            downloadedPendingIntent);
}
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/MmsSender.java

</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

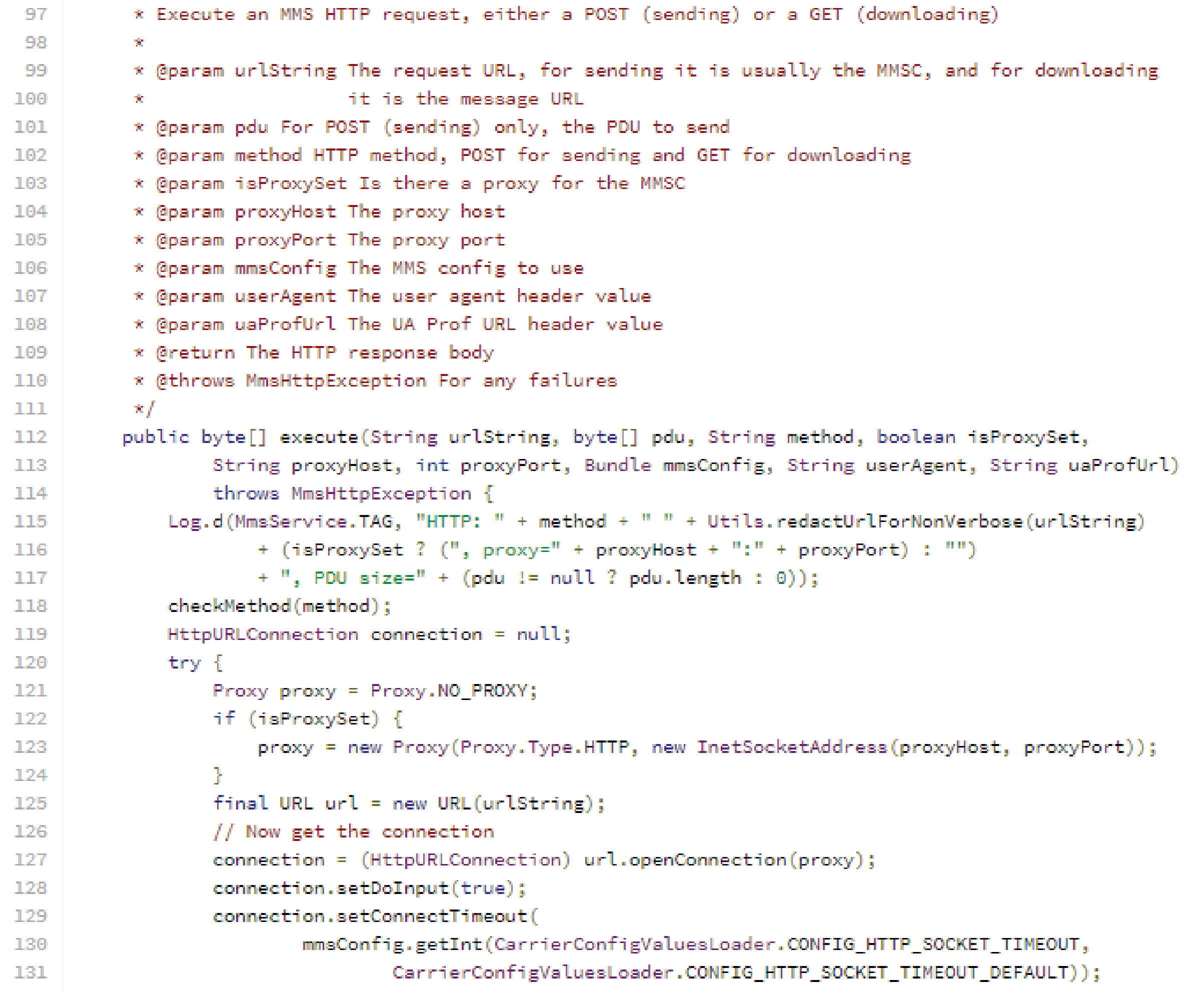
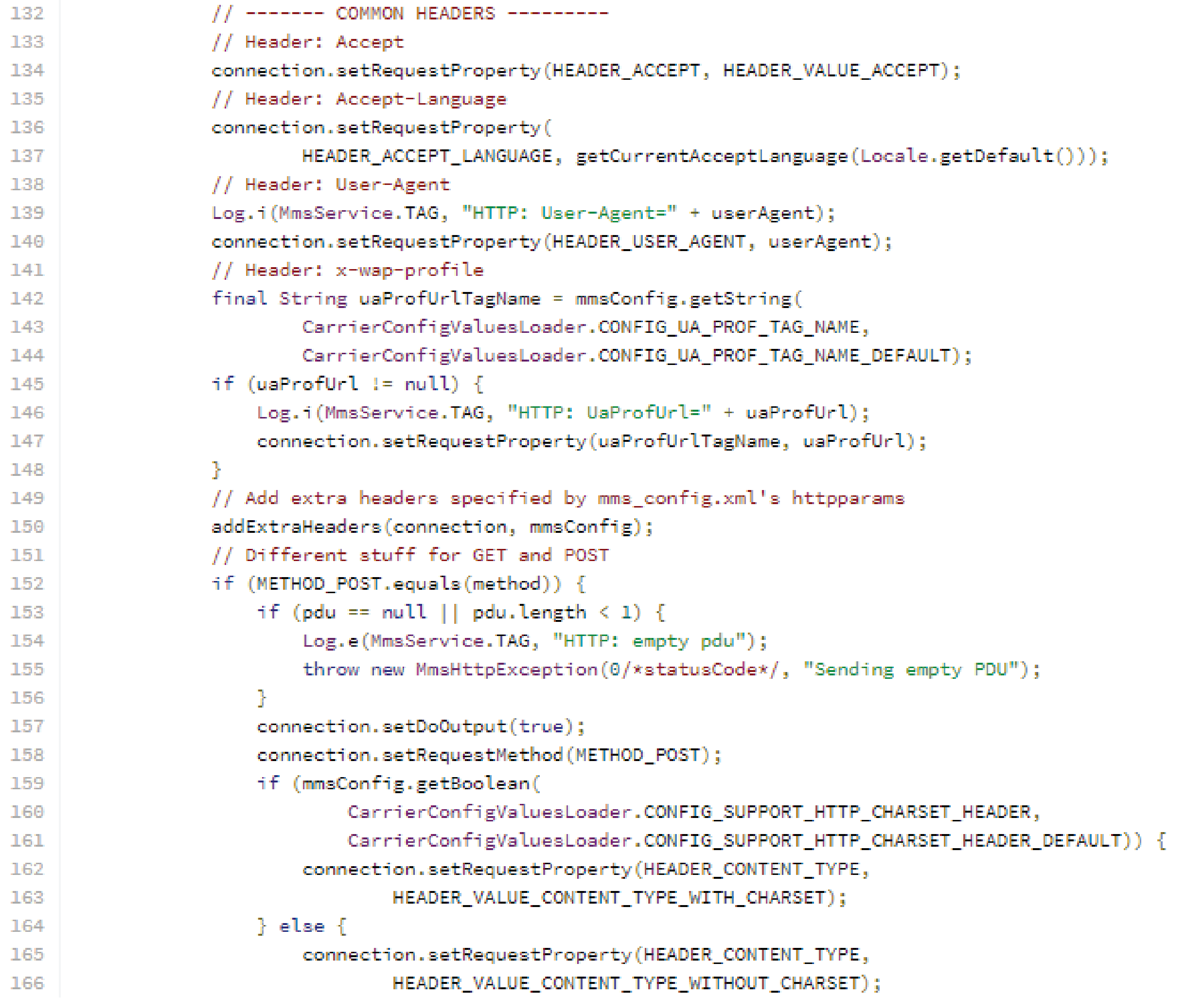
<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>
<pre>
     * Execute an MMS HTTP request, either a POST (sending) or a GET (downloading)
     *
     * @param urlString The request URL, for sending it is usually the MMSC, and for downloading
     *   it is the message URL
     * @param pdu For POST (sending) only, the PDU to send
     * @param method HTTP method, POST for sending and GET for downloading
     * @param isProxySet Is there a proxy for the MMSC
     * @param proxyHost The proxy host
     * @param proxyPort The proxy port
     * @param mmsConfig The MMS config to use
     * @param userAgent The user agent header value
     * @param uaProfUrl The UA Prof URL header value
     * @return The HTTP response body
     * @throws MmsHttpException For any failures
     */
    public byte[] execute(String urlString, byte[] pdu, String method, boolean isProxySet,
            String proxyHost, int proxyPort, Bundle mmsConfig, String userAgent, String uaProfUrl)
            throws MmsHttpException {
        Log.d(MmsService.TAG, "HTTP: " + method + " " + Utils.redactUrlForNonVerbose(urlString)
                + (isProxySet ? (", proxy=" + proxyHost + ":" + proxyPort) : "")
                + ", PDU size=" + (pdu != null ? pdu.length : 0));
        checkMethod(method);
        HttpURLConnection connection = null;
        try {
            Proxy proxy = Proxy.NO_PROXY;
            if (isProxySet) {
                proxy = new Proxy(Proxy.Type.HTTP, new InetSocketAddress(proxyHost, proxyPort));
            }
            final URL url = new URL(urlString);
            // Now get the connection
            connection = (HttpURLConnection) url.openConnection(proxy);
            connection.setDoInput(true);
            connection.setConnectTimeout(
                    mmsConfig.getInt(CarrierConfigValuesLoader.CONFIG_HTTP_SOCKET_TIMEOUT,
                            CarrierConfigValuesLoader.CONFIG_HTTP_SOCKET_TIMEOUT_DEFAULT));
</pre>
https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/android/support/v7/mms/MmsHttpClient.java</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

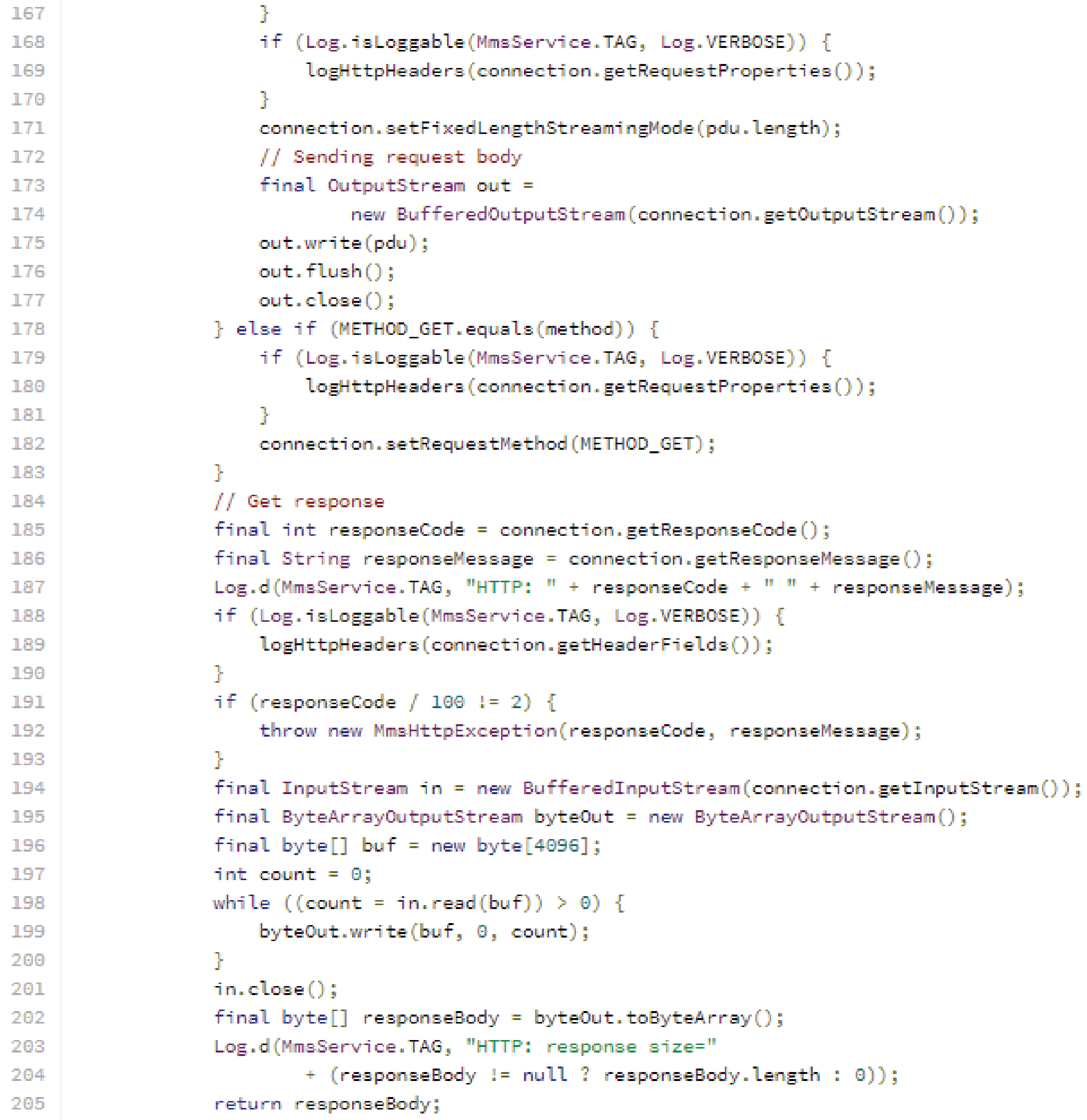
US9467838	Exemplary Supporting Evidence Regarding Accused Products



```
            // ------- COMMON HEADERS ---------
            // Header: Accept
            connection.setRequestProperty(HEADER_ACCEPT, HEADER_VALUE_ACCEPT);
            // Header: Accept-Language
            connection.setRequestProperty(
                    HEADER_ACCEPT_LANGUAGE, getCurrentAcceptLanguage(Locale.getDefault()));
            // Header: User-Agent
            Log.i(MmsService.TAG, "HTTP: User-Agent=" + userAgent);
            connection.setRequestProperty(HEADER_USER_AGENT, userAgent);
            // Header: x-wap-profile
            final String uaProfUrlTagName = mmsConfig.getString(
                    CarrierConfigValuesLoader.CONFIG_UA_PROF_TAG_NAME,
                    CarrierConfigValuesLoader.CONFIG_UA_PROF_TAG_NAME_DEFAULT);
            if (uaProfUrl != null) {
                Log.i(MmsService.TAG, "HTTP: UaProfUrl=" + uaProfUrl);
                connection.setRequestProperty(uaProfUrlTagName, uaProfUrl);
            }
            // Add extra headers specified by mms_config.xml's httpparams
            addExtraHeaders(connection, mmsConfig);
            // Different stuff for GET and POST
            if (METHOD_POST.equals(method)) {
                if (pdu == null || pdu.length < 1) {
                    Log.e(MmsService.TAG, "HTTP: empty pdu");
                    throw new MmsHttpException(0/*statusCode*/, "Sending empty PDU");
                }
                connection.setDoOutput(true);
                connection.setRequestMethod(METHOD_POST);
                if (mmsConfig.getBoolean(
                        CarrierConfigValuesLoader.CONFIG_SUPPORT_HTTP_CHARSET_HEADER,
                        CarrierConfigValuesLoader.CONFIG_SUPPORT_HTTP_CHARSET_HEADER_DEFAULT)) {
                    connection.setRequestProperty(HEADER_CONTENT_TYPE,
                            HEADER_VALUE_CONTENT_TYPE_WITH_CHARSET);
                } else {
                    connection.setRequestProperty(HEADER_CONTENT_TYPE,
                            HEADER_VALUE_CONTENT_TYPE_WITHOUT_CHARSET);
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/android/support/v7/mms/MmsHttpClient.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

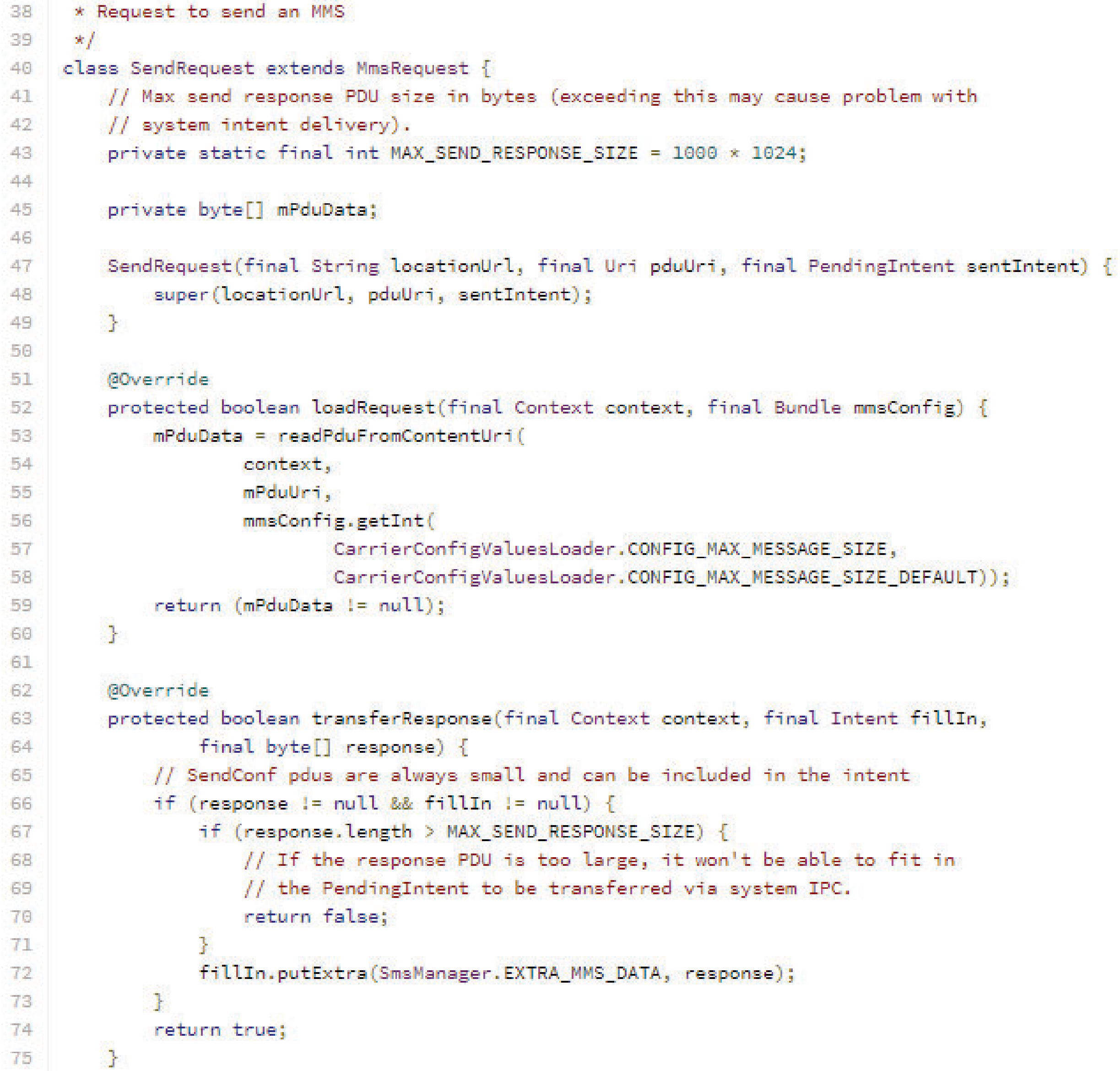
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
            }
            if (Log.isLoggable(MmsService.TAG, Log.VERBOSE)) {
                logHttpHeaders(connection.getRequestProperties());
            }
            connection.setFixedLengthStreamingMode(pdu.length);
            // Sending request body
            final OutputStream out =
                    new BufferedOutputStream(connection.getOutputStream());
            out.write(pdu);
            out.flush();
            out.close();
        } else if (METHOD_GET.equals(method)) {
            if (Log.isLoggable(MmsService.TAG, Log.VERBOSE)) {
                logHttpHeaders(connection.getRequestProperties());
            }
            connection.setRequestMethod(METHOD_GET);
        }
        // Get response
        final int responseCode = connection.getResponseCode();
        final String responseMessage = connection.getResponseMessage();
        Log.d(MmsService.TAG, "HTTP: " + responseCode + " " + responseMessage);
        if (Log.isLoggable(MmsService.TAG, Log.VERBOSE)) {
            logHttpHeaders(connection.getHeaderFields());
        }
        if (responseCode / 100 != 2) {
            throw new MmsHttpException(responseCode, responseMessage);
        }
        final InputStream in = new BufferedInputStream(connection.getInputStream());
        final ByteArrayOutputStream byteOut = new ByteArrayOutputStream();
        final byte[] buf = new byte[4096];
        int count = 0;
        while ((count = in.read(buf)) > 0) {
            byteOut.write(buf, 0, count);
        }
        in.close();
        final byte[] responseBody = byteOut.toByteArray();
        Log.d(MmsService.TAG, "HTTP: response size="
                + (responseBody != null ? responseBody.length : 0));
        return responseBody;
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	release/src/android/support/v7/mms/MmsHttpClient.java

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>



```
 * Request to send an MMS
 */
class SendRequest extends MmsRequest {
    // Max send response PDU size in bytes (exceeding this may cause problem with
    // system intent delivery).
    private static final int MAX_SEND_RESPONSE_SIZE = 1000 * 1024;

    private byte[] mPduData;

    SendRequest(final String locationUrl, final Uri pduUri, final PendingIntent sentIntent) {
        super(locationUrl, pduUri, sentIntent);
    }

    @Override
    protected boolean loadRequest(final Context context, final Bundle mmsConfig) {
        mPduData = readPduFromContentUri(
                context,
                mPduUri,
                mmsConfig.getInt(
                        CarrierConfigValuesLoader.CONFIG_MAX_MESSAGE_SIZE,
                        CarrierConfigValuesLoader.CONFIG_MAX_MESSAGE_SIZE_DEFAULT));
        return (mPduData != null);
    }

    @Override
    protected boolean transferResponse(final Context context, final Intent fillIn,
            final byte[] response) {
        // SendConf pdus are always small and can be included in the intent
        if (response != null && fillIn != null) {
            if (response.length > MAX_SEND_RESPONSE_SIZE) {
                // If the response PDU is too large, it won't be able to fit in
                // the PendingIntent to be transferred via system IPC.
                return false;
            }
            fillIn.putExtra(SmsManager.EXTRA_MMS_DATA, response);
        }
        return true;
    }
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/android/support/v7/mms/SendRequest.java

</td></tr>
</table>

US9467838	Exemplary Supporting Evidence Regarding Accused Products
[1C] participating in the group, wherein participating in the group includes sending first location information to a first server and receiving second location information from the first server, the first location information comprising a location of the first device, the second location information comprising one or more locations of one or more respective second devices included in the group;	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of: participating in the group, wherein participating in the group includes sending first location information to a first server and receiving second location information from the first server, the first location information comprising a location of the first device, the second location information comprising one or more locations of one or more respective second devices included in the group. **Regarding Find My Device** and Android Device Manager, the Accused Products require a user to join the corresponding network by: signing-in to the device with an identifier (e.g., Google Account) or linking the device to the identifier by remote means. The Accused Products require that a signed-in user of a first device to share its location by enabling Location Access or Location Services. When a user of a first device associated with an identity (e.g. Google Account) enables the Find My Device feature and Location Services on the first device, the user shares its location and the first device's location is sent to a server. When the user makes a request to view a map of device locations associated with the identity, the device receives one or more locations corresponding to one or more second devices associated with the identity. Alternatively, the locations corresponding to one or more second devices associated with the identity are sent to the first device on a rolling basis. **Regarding Google Maps**, Google Latitude, Google Plus, Google Hangouts, Google Messages, Google Allo, Google Duo, Google Chrome, and Android Messenger, the Accused Products require a user to join the corresponding network by signing-in to the device with an identifier (e.g., Google Account). When the signed-in user enables Google Maps Share Location and Location Services on the first device, the user shares its location and the first device's location is sent to a server. When the user enables sharing to one or more contacts (of respective devices) and the one or more contacts enable sharing their location to the user of the first device, the user of the first device receives the locations of the one or more contacts. **Further regarding Google Maps**, Google Latitude, Google Plus, Google Hangouts, Google Messages, Google Allo, Google Duo, Google Chrome, and Android Messenger, the Accused Products alternatively require a user to send a request containing the identifier (e.g., Google Account). When the signed-in user enables Google Maps Share Location and Location Services on the first device, the user shares its location and the first device's location is sent to a server. When the user sends a message to another contact through Google Maps, Google Messages, and/or another means from within the Google Maps application, the message including location information are sent to a server before transmission to the intended contact.

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

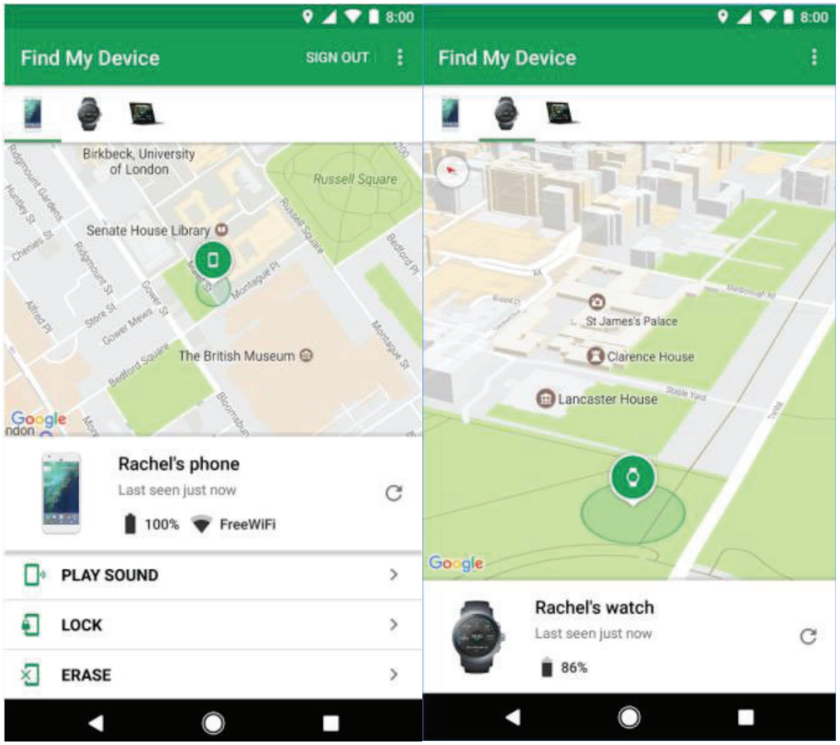
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	When one or more contacts enable sharing their location to the user of the first device, or alternatively send a message containing location information, or alternatively accept a request to share their location with the first user, the user of the first device receives the locations of the one or more contacts. **<u>Exemplary Support for Find My Device</u>**:  8:00 Find My Device SIGN OUT Birkbeck, University of London Russell Square Senate House Library The British Museum Google Rachel's phone Last seen just now 100% FreeWiFi PLAY SOUND LOCK ERASE 8:00 Find My Device St James's Palace Clarence House Lancaster House Google Rachel's watch Last seen just now 86% https://play.google.com/store/apps/details?id=com.google.android.apps.adm&hl=en

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

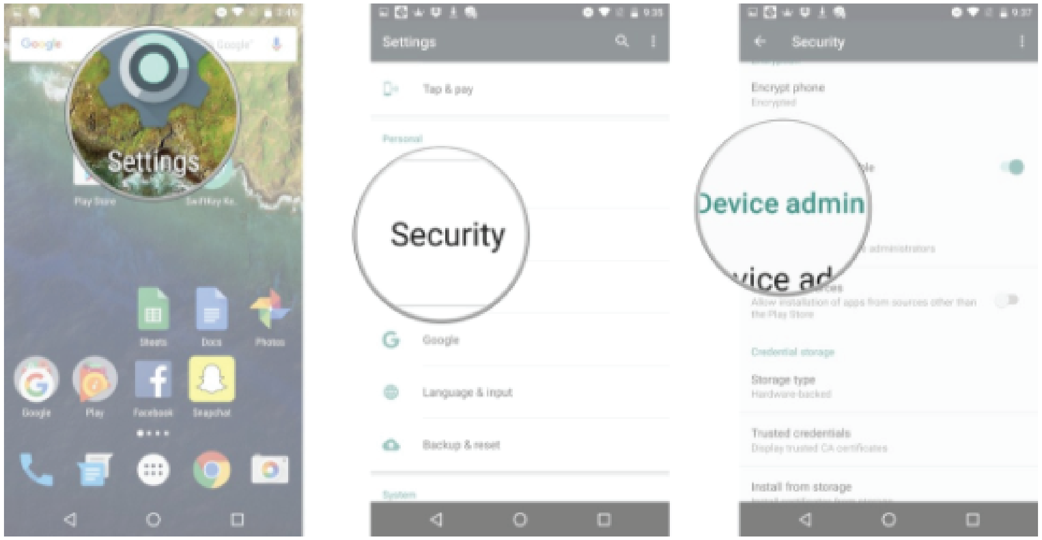
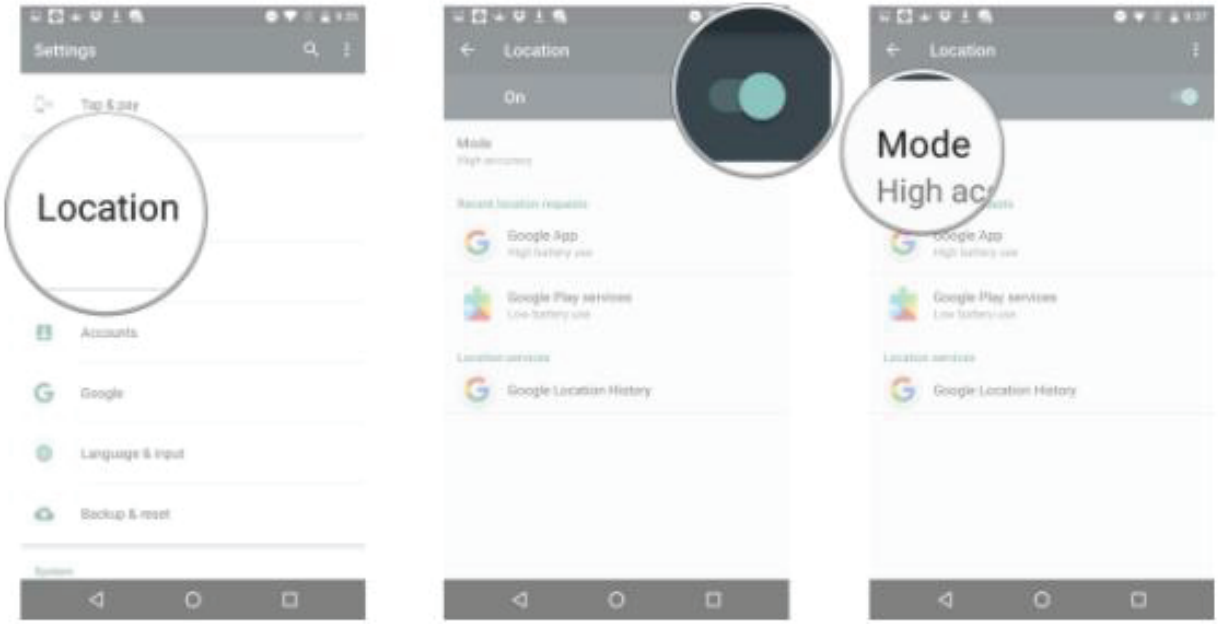
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to enable Find My Device on your phone In newer Android phones, the Find My Device service is already located conveniently in your Settings app, but if you can't find it you can always download Find My Device from the Google Play Store. This locating service has essentially amalgamated with Google to make finding your phone easier. There are just a couple of things you'll need to activate. 1. Launch **Settings**. 2. Tap **Security**. 3. Tap **Device Administration**.  Settings Security Device admin Tap & pay Google Language & input Backup & reset Encrypt phone Storage type Trusted credentials Install from storage 4. Tap **Find My Device** so that a checkmark appears in the checkbox. https://www.androidcentral.com/how-track-android-phone

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	7. Tap **Location** in the main Settings menu. 8. Tap the **switch** beside **Location** at the top of the screen so that it turns on. 9. Tap **Mode**.  Settings Tap & pay Location Accounts Google Language & input Backup & reset System Location On Mode High accuracy Recent location requests Google App High battery use Google Play services Low battery use Location services Google Location History Location Mode High ac https://www.androidcentral.com/how-track-android-phone

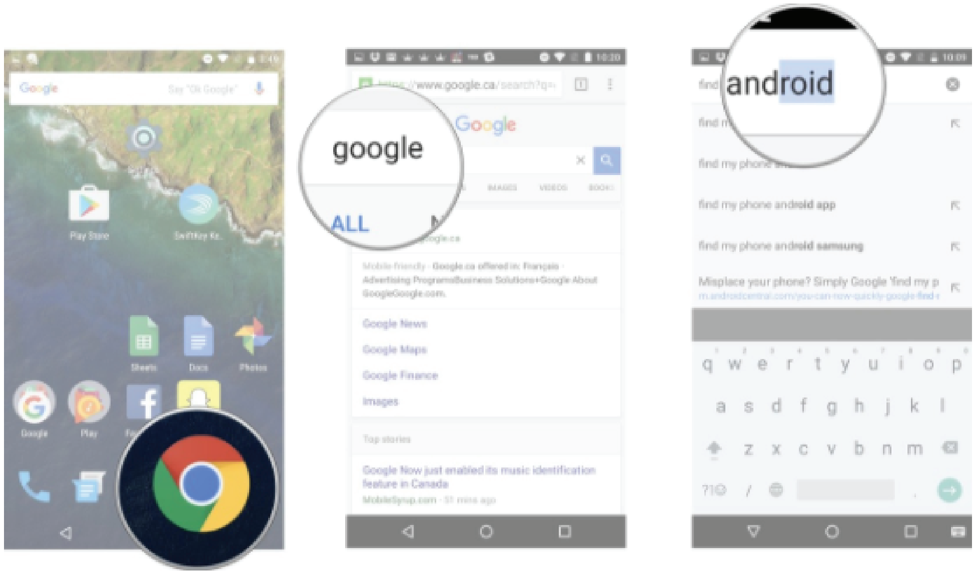
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to locate your phone with Google Should you happen to lose your phone, you can locate its whereabouts by logging into your Google account from any computer or even from another phone. 1. Launch a **web browser** from a phone, tablet, or computer. 2. Navigate to **Google** if it is not your default search engine or home page. 3. Type **find my phone android** in the Google search bar.  google ALL android find my phone android app find my phone android samsung 4. Tap on **Find My Device** (usually the first option in the search). 5. Enter your **email address** and **password** just as though you were checking your email. If you have 2-step verification set up on your Google account (and you most certainly should), you'll need to complete that process as well. https://www.androidcentral.com/how-track-android-phone

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

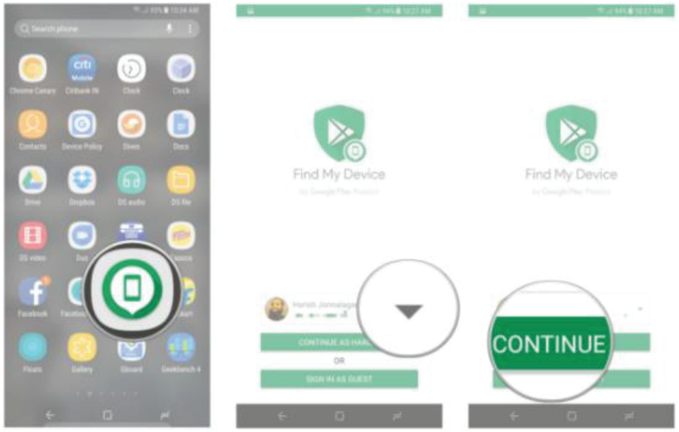
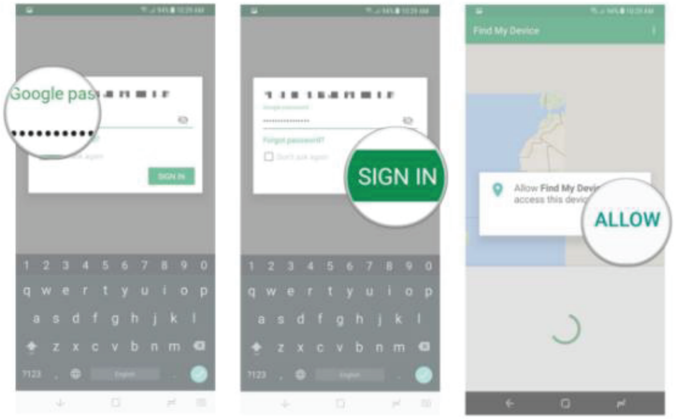
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	1. Open **Find Device** from your home screen or app drawer. 2. Select the **Google account** you want to use the service with. 3. Hit the **Continue as** button.  4. Enter your **Google account password**. 5. Tap **Sign in**. 6. Give **location access** to the service.  https://www.androidcentral.com/find-my-device

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

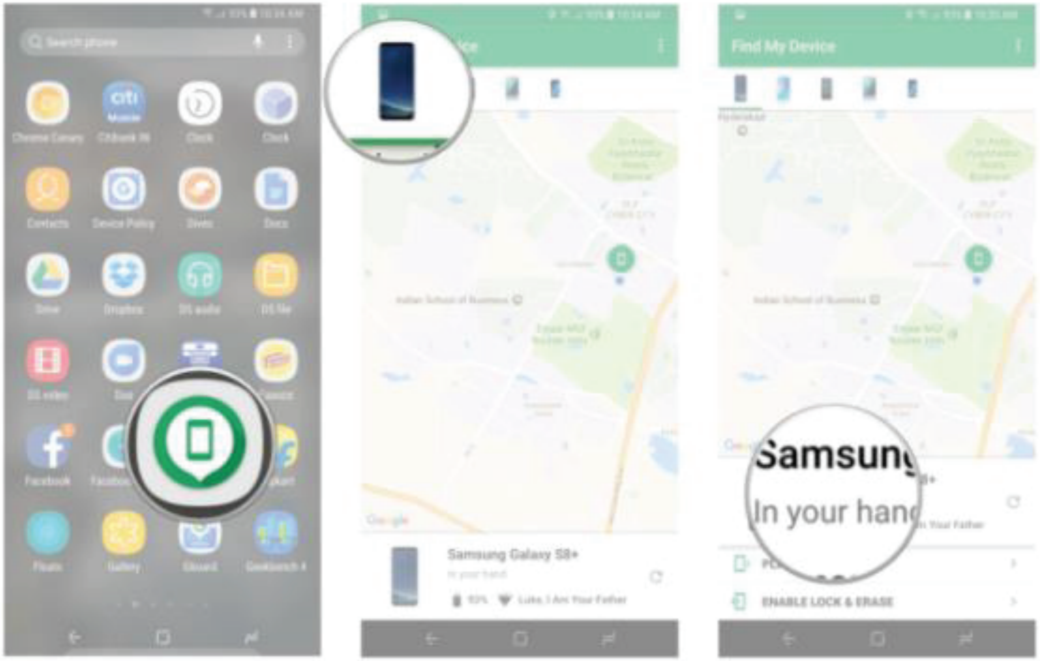
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	Once you're signed in to Find My Device, you'll see a map with your current location as well as the make and model of your phone, and two options — Play Sound, and Enable Lock & Erase. Hitting the latter option will allow you to start using the Lock and Erase functions. If you've signed into more than one phone, you can select a particular device by browsing the list at the top of the screen. 1. Open **Find My Device** from your home screen or app drawer. 2. Select your phone from the **list of devices at the top of the screen**. 3. See if your phone is **discoverable**.  Find My Device Samsung Galaxy S8+ Samsung In your hand ENABLE LOCK & ERASE https://www.androidcentral.com/find-my-device

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

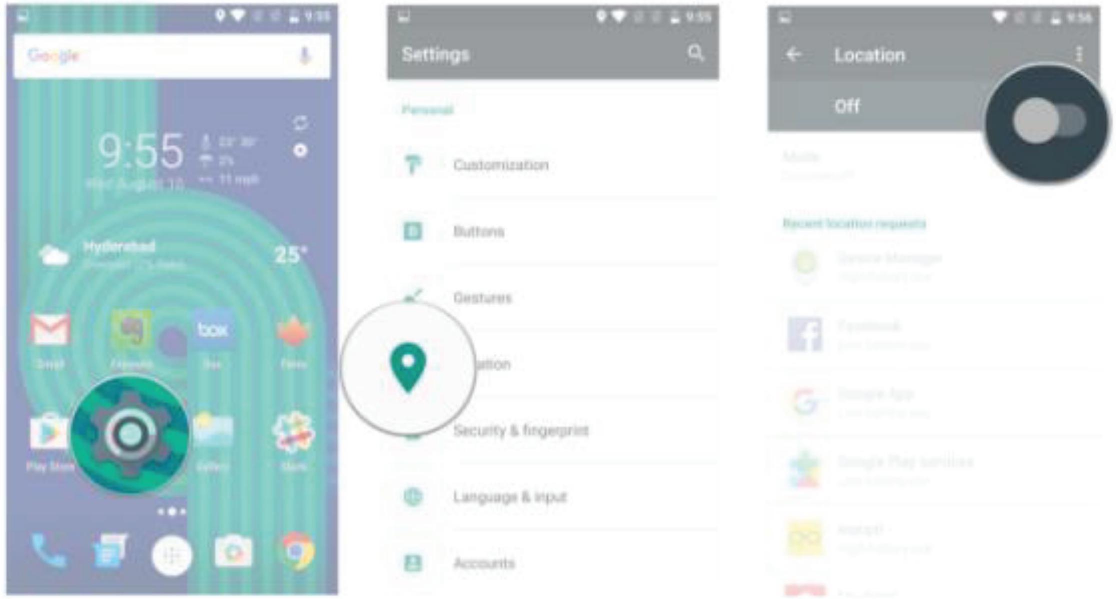
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	If you're not able to find your phone or if it says that the device is unavailable, it is likely that the location services are disabled. Find My Device relies on GPS to track your phone, so now would be a good time to enable location services. 1. Open **Settings** from your home screen or app drawer. 2. Tap **Location**. 3. Toggle **Enable location services**.  9:55 Settings Personal Customization Buttons Gestures …ation Security & fingerprint Language & input Accounts Location Off Recent location requests https://www.androidcentral.com/find-my-device **Exemplary Support for Google Maps**:

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

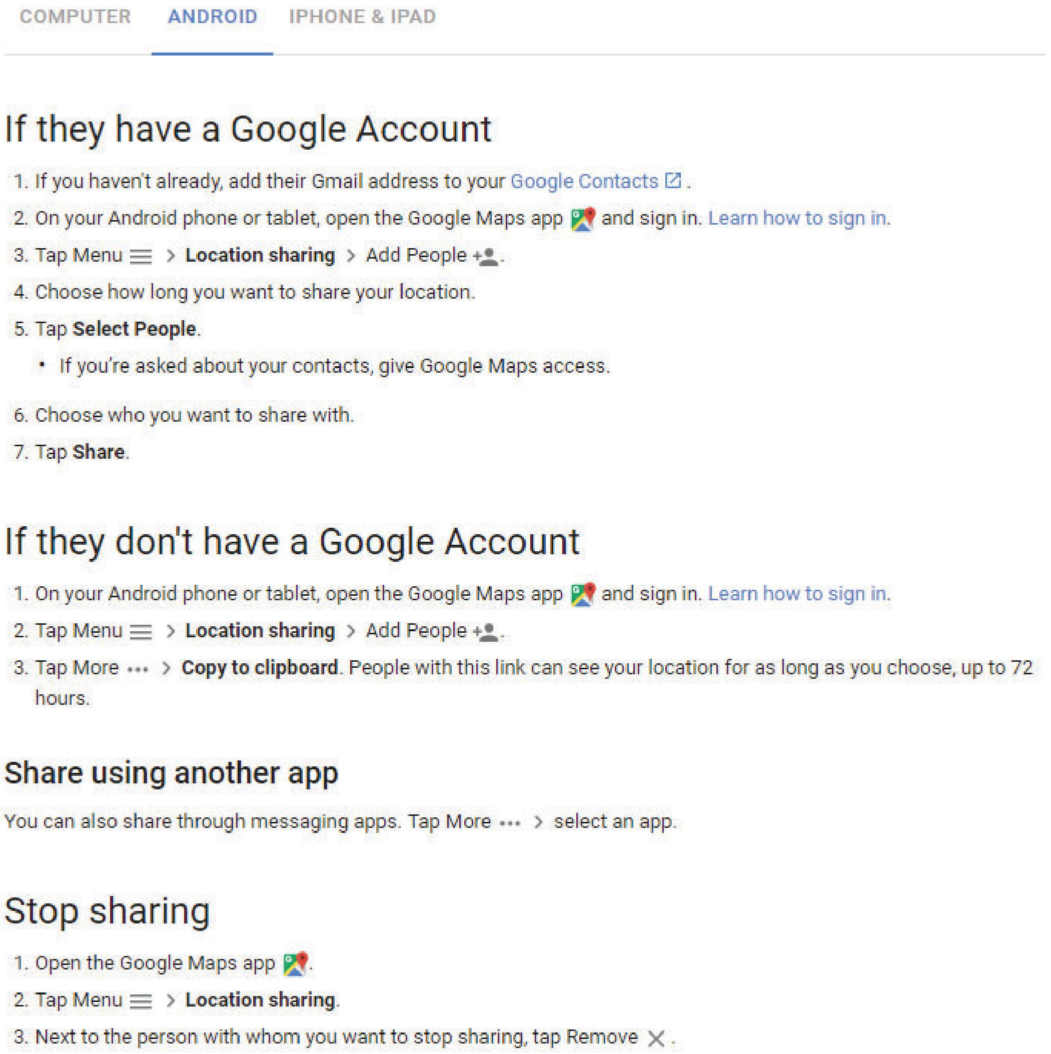
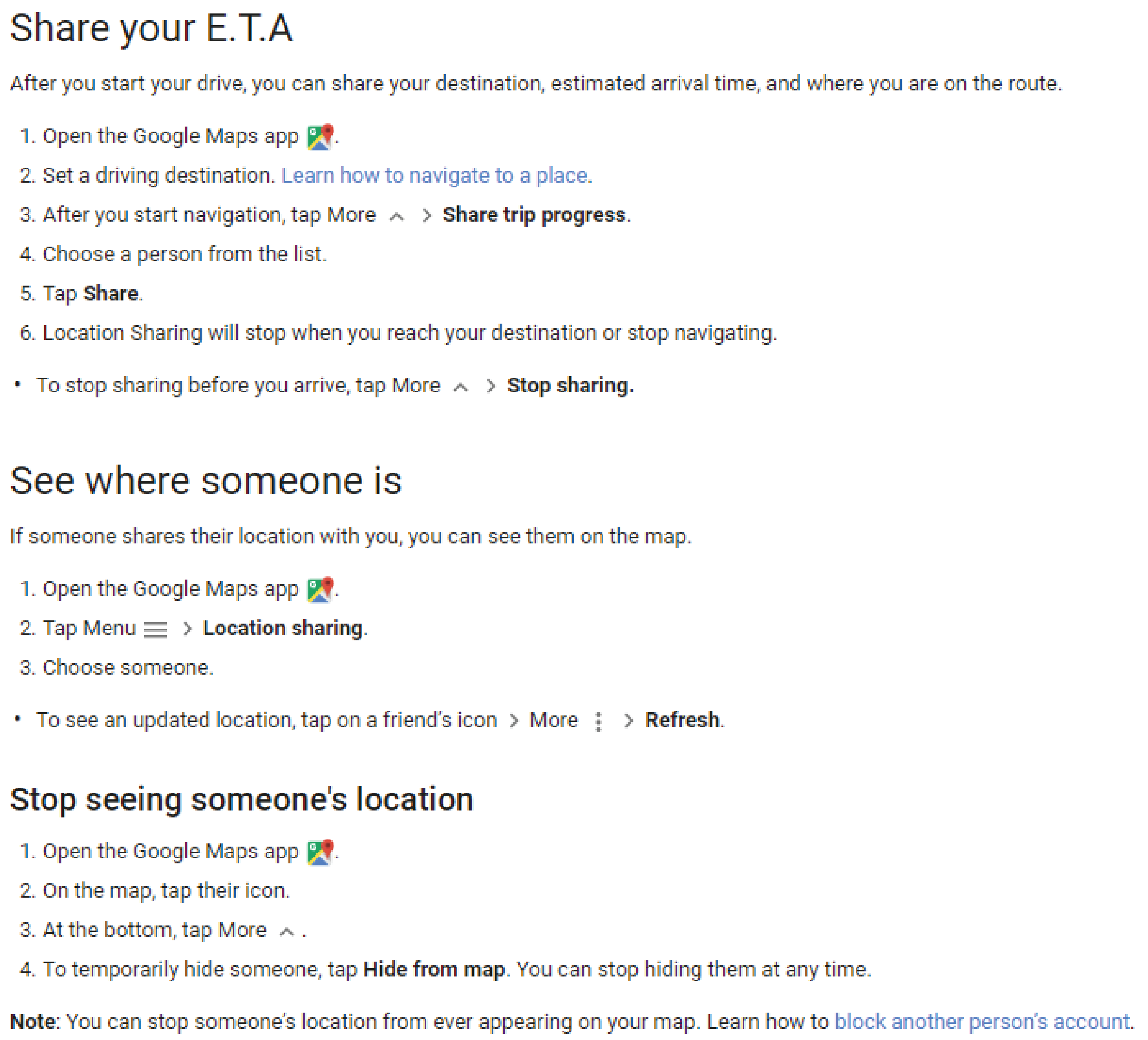
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 COMPUTER ANDROID IPHONE & IPAD If they have a Google Account 1. If you haven't already, add their Gmail address to your Google Contacts ↗. 2. On your Android phone or tablet, open the Google Maps app and sign in. Learn how to sign in. 3. Tap Menu ≡ > **Location sharing** > Add People +👤. 4. Choose how long you want to share your location. 5. Tap **Select People**. - If you're asked about your contacts, give Google Maps access. 6. Choose who you want to share with. 7. Tap **Share**. If they don't have a Google Account 1. On your Android phone or tablet, open the Google Maps app and sign in. Learn how to sign in. 2. Tap Menu ≡ > **Location sharing** > Add People +👤. 3. Tap More ••• > **Copy to clipboard**. People with this link can see your location for as long as you choose, up to 72 hours. Share using another app You can also share through messaging apps. Tap More ••• > select an app. Stop sharing 1. Open the Google Maps app. 2. Tap Menu ≡ > **Location sharing**. 3. Next to the person with whom you want to stop sharing, tap Remove ✕. https://support.google.com/maps/answer/7326816?co=GENIE.Platform%3DAndroid&oco=1

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Share your E.T.A After you start your drive, you can share your destination, estimated arrival time, and where you are on the route. 1. Open the Google Maps app. 2. Set a driving destination. Learn how to navigate to a place. 3. After you start navigation, tap More ^ > **Share trip progress**. 4. Choose a person from the list. 5. Tap **Share**. 6. Location Sharing will stop when you reach your destination or stop navigating. • To stop sharing before you arrive, tap More ^ > **Stop sharing.** See where someone is If someone shares their location with you, you can see them on the map. 1. Open the Google Maps app. 2. Tap Menu ☰ > **Location sharing**. 3. Choose someone. • To see an updated location, tap on a friend's icon > More ⋮ > **Refresh**. Stop seeing someone's location 1. Open the Google Maps app. 2. On the map, tap their icon. 3. At the bottom, tap More ^ . 4. To temporarily hide someone, tap **Hide from map**. You can stop hiding them at any time. **Note**: You can stop someone's location from ever appearing on your map. Learn how to block another person's account. https://support.google.com/maps/answer/7326816?co=GENIE.Platform%3DAndroid&oco=1

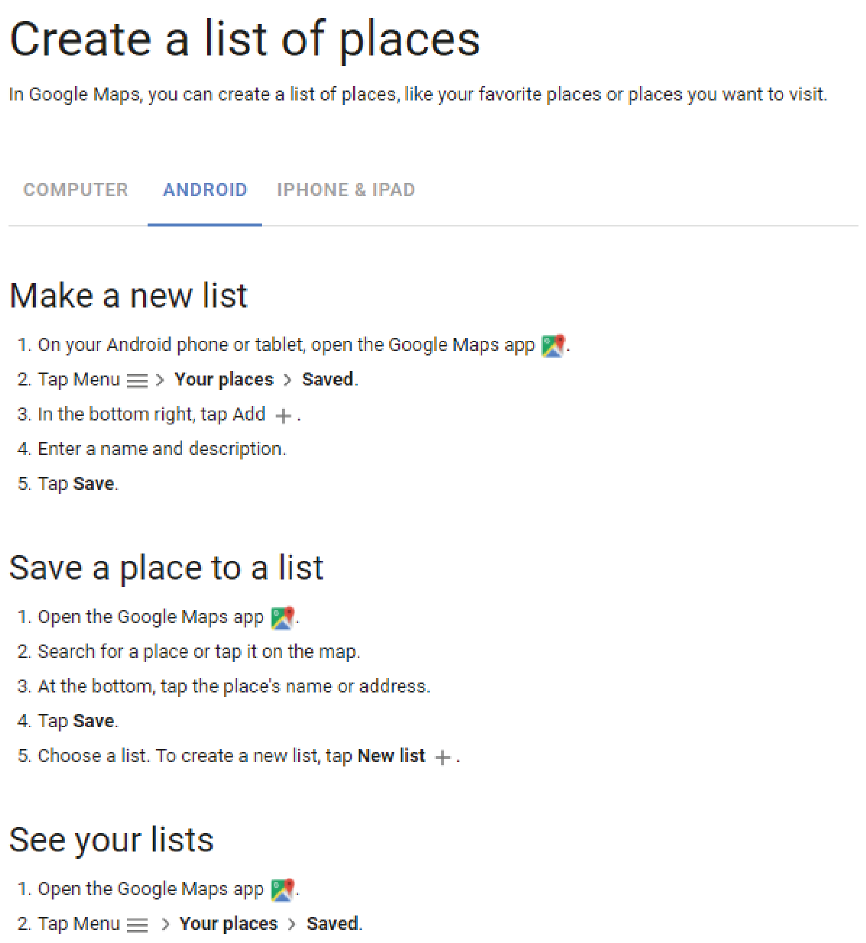
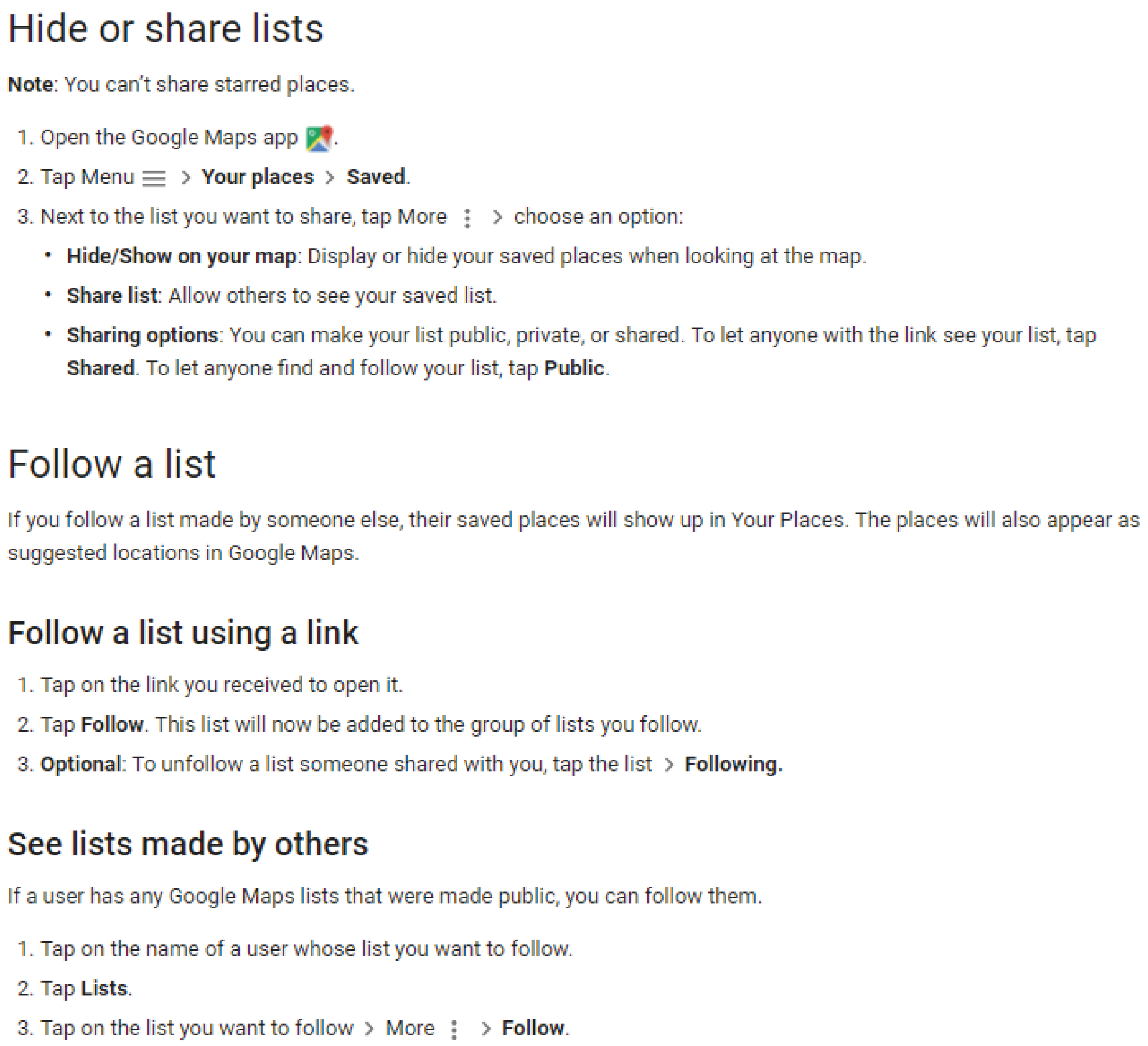
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Create a list of places In Google Maps, you can create a list of places, like your favorite places or places you want to visit. COMPUTER ANDROID IPHONE & IPAD Make a new list 1. On your Android phone or tablet, open the Google Maps app. 2. Tap Menu ☰ > **Your places** > **Saved**. 3. In the bottom right, tap Add +. 4. Enter a name and description. 5. Tap **Save**. Save a place to a list 1. Open the Google Maps app. 2. Search for a place or tap it on the map. 3. At the bottom, tap the place's name or address. 4. Tap **Save**. 5. Choose a list. To create a new list, tap **New list** +. See your lists 1. Open the Google Maps app. 2. Tap Menu ☰ > **Your places** > **Saved**. https://support.google.com/maps/answer/7280933?hl=en&ref_topic=7301134&co=GENIE.Platform%3DAndroid&oco=1

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 **Hide or share lists** **Note**: You can't share starred places. 1. Open the Google Maps app. 2. Tap Menu ≡ > **Your places** > **Saved**. 3. Next to the list you want to share, tap More ⋮ > choose an option: • **Hide/Show on your map**: Display or hide your saved places when looking at the map. • **Share list**: Allow others to see your saved list. • **Sharing options**: You can make your list public, private, or shared. To let anyone with the link see your list, tap **Shared**. To let anyone find and follow your list, tap **Public**. **Follow a list** If you follow a list made by someone else, their saved places will show up in Your Places. The places will also appear as suggested locations in Google Maps. **Follow a list using a link** 1. Tap on the link you received to open it. 2. Tap **Follow**. This list will now be added to the group of lists you follow. 3. **Optional**: To unfollow a list someone shared with you, tap the list > **Following.** **See lists made by others** If a user has any Google Maps lists that were made public, you can follow them. 1. Tap on the name of a user whose list you want to follow. 2. Tap **Lists**. 3. Tap on the list you want to follow > More ⋮ > **Follow**. https://support.google.com/maps/answer/7280933?hl=en&ref_topic=7301134&co=GENIE.Platform%3DAndroid&oco=1

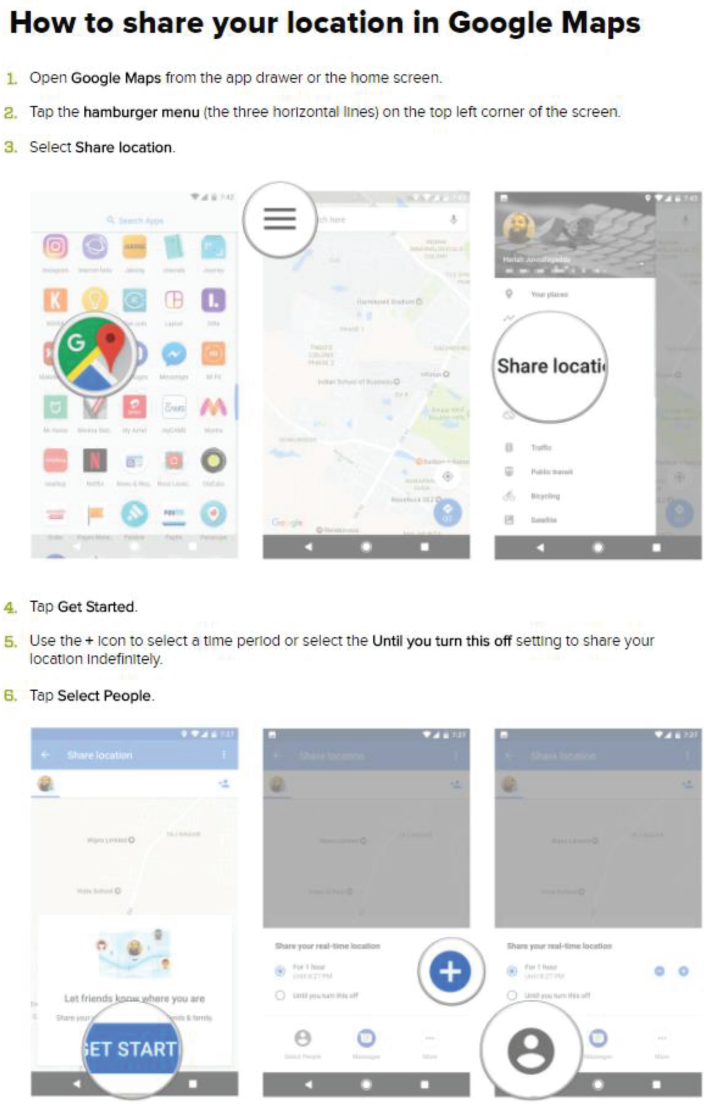
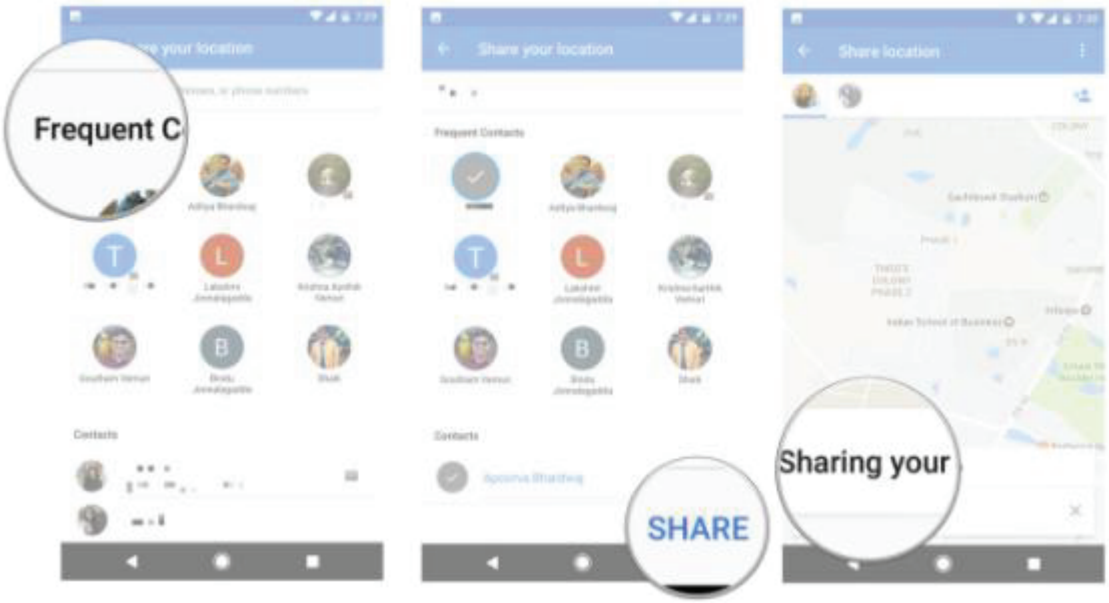
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 **How to share your location in Google Maps** 1. Open **Google Maps** from the app drawer or the home screen. 2. Tap the **hamburger menu** (the three horizontal lines) on the top left corner of the screen. 3. Select **Share location**. 4. Tap **Get Started**. 5. Use the **+** icon to select a time period or select the **Until you turn this off** setting to share your location indefinitely. 6. Tap **Select People**. Share location Let friends know where you are ET START Share your real-time location For 1 hour Until you turn this off https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	7. You'll see a list of your frequent contacts at the top, along with a full list of contacts. **Pick the contacts** by tapping their name. 8. Once you've selected the contacts you want to share your location to, tap **Share**. 9. You'll see a message saying that the selected contact can view your location.  Frequent C Share your location Frequent Contacts Contacts SHARE Share location Sharing your https://www.androidcentral.com/how-share-location-google-maps

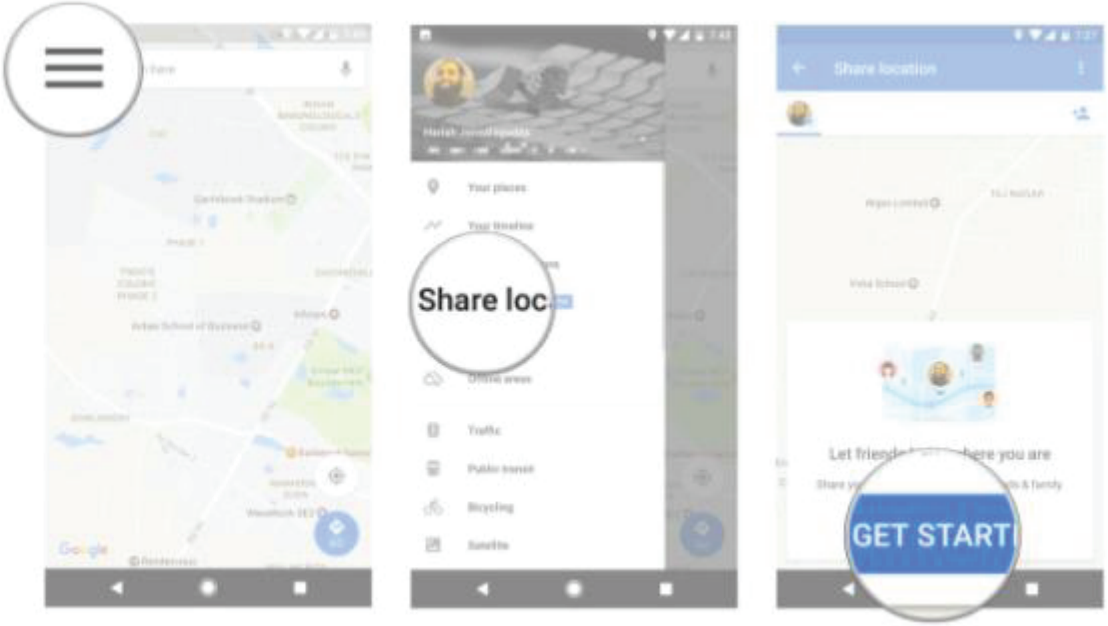
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to create a shareable link You can also create a link and use it to share your location easily. Here's how to do it: 1. Tap the **hamburger menu** on the top left corner of the screen. 2. Select **Share location**. 3. Tap **Get Started**.  Share location Your places Your timeline Share loc Traffic Public transit Bicycling Satellite Share location Let friends ... where you are GET START https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

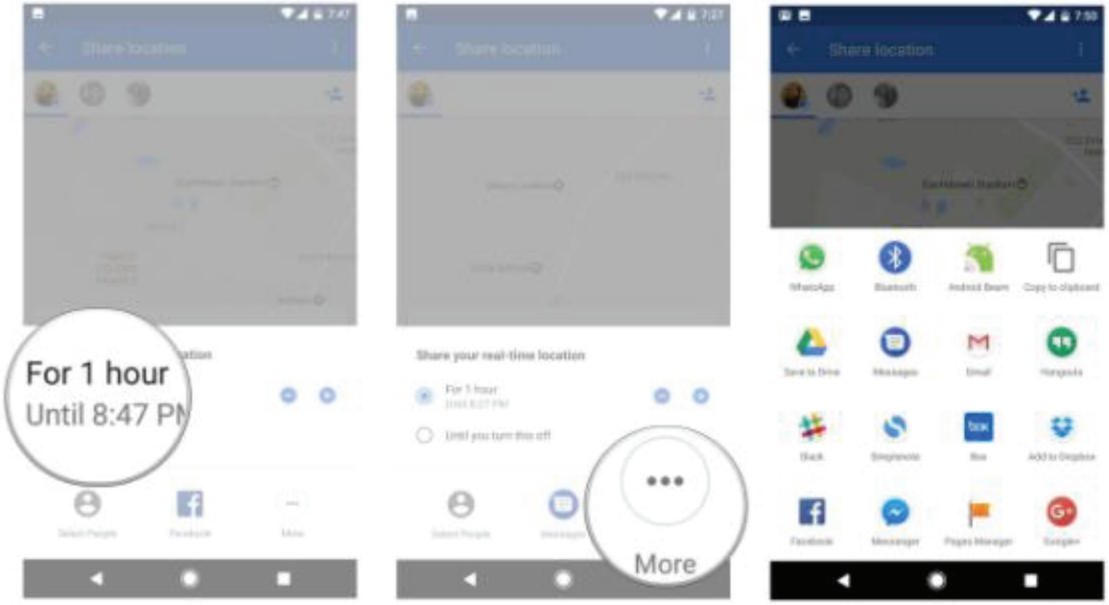
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	4. Select the **amount of time** you want to share your location. 5. Tap **More**. 6. Select **your app of choice** to create and send a unique URL that broadcasts your current location. You can email it, send the link via Messenger, or even tweet it to the intended recipient.  Share location For 1 hour Until 8:47 PM Share your real-time location For 1 hour Until you turn this off More WhatsApp Bluetooth Android Beam Copy to clipboard Save to Drive Messages Email Hangouts Slack Simplenote Box Add to Dropbox Facebook Messenger Pages Manager Google+ https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

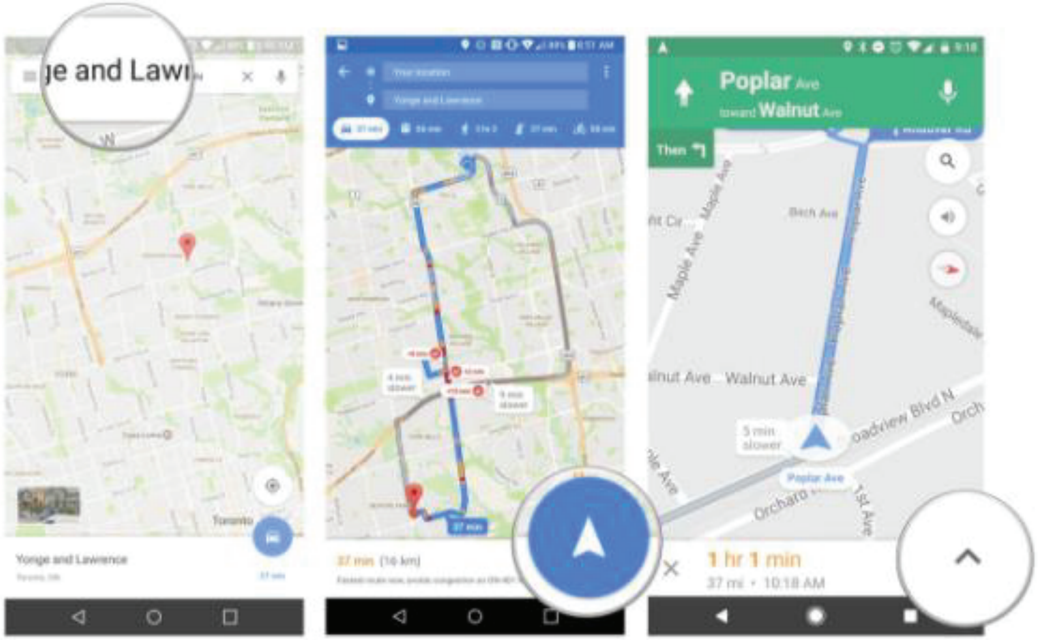
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to share your navigation directions while you walk, drive or transit One of the best ways to share your location with someone is during a drive, walk or transit. If you're meeting a friend or family member somewhere, or navigating towards their home, Google Maps lets you share your location with them for the duration of the trip. It's magic! 1. In the **search bar** enter your destination. 2. Pick your navigation type (drive, transit, walk) and press the blue **navigate button**. 3. Tap the arrow next to the **time-to-destination** number at the bottom of the screen.  je and Lawr Yonge and Lawrence Toronto 37 min (16 km) Poplar Ave toward Walnut Ave Then Maple Ave Birch Ave Walnut Ave 5 min slower Poplar Ave Orchard 1st Ave 1 hr 1 min 37 mi • 10:18 AM https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

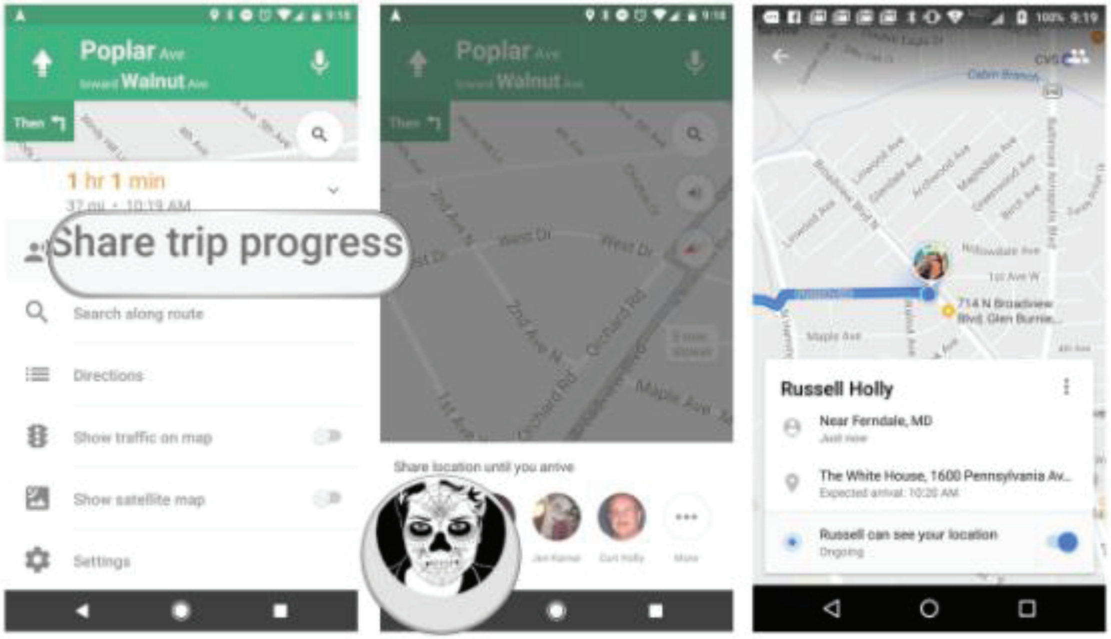
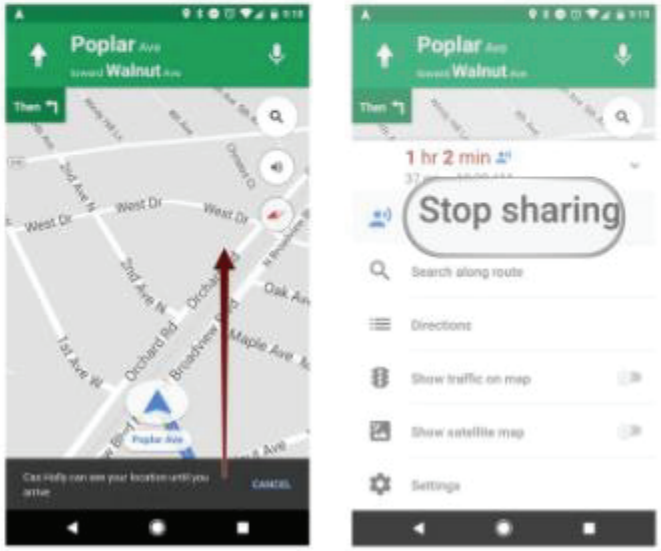
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	4. Tap **Share trip progress**. 5. Choose one or more contacts to share trip progress.  Poplar Ave toward Walnut Ave Then 1 hr 1 min 37 mi · 10:19 AM Share trip progress Search along route Directions Show traffic on map Show satellite map Settings Share location until you arrive More Russell Holly Near Ferndale, MD Just now The White House, 1600 Pennsylvania Av... Expected arrival: 10:20 AM Russell can see your location Ongoing 714 N Broadview Blvd, Glen Burnie... You can also stop sharing your location with someone before a trip ends. https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	1. Tap the arrow next to the **time-to-destination** number at the bottom of the screen. 2. Tap **Stop sharing**.  Poplar Ave toward Walnut Ave Then West Dr 2nd Ave N Orchard Rd Broadview Maple Ave Oak Ave 1st Ave W Poplar Ave Can Holly can see your location until you arrive CANCEL 1 hr 2 min Stop sharing Search along route Directions Show traffic on map Show satellite map Settings That's it! Are you excited that location sharing is back in Google Maps? How often do you use the feature? https://www.androidcentral.com/how-share-location-google-maps As shown below, a group may also be defined within Google Contacts.

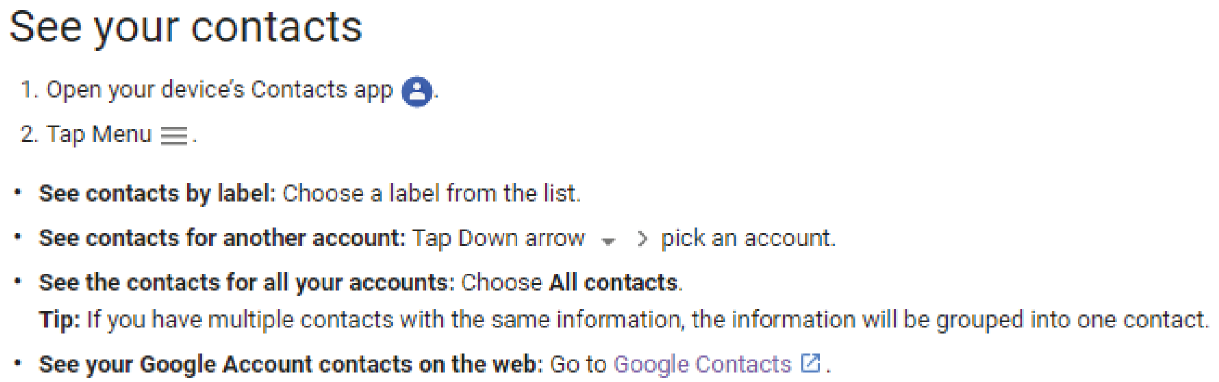
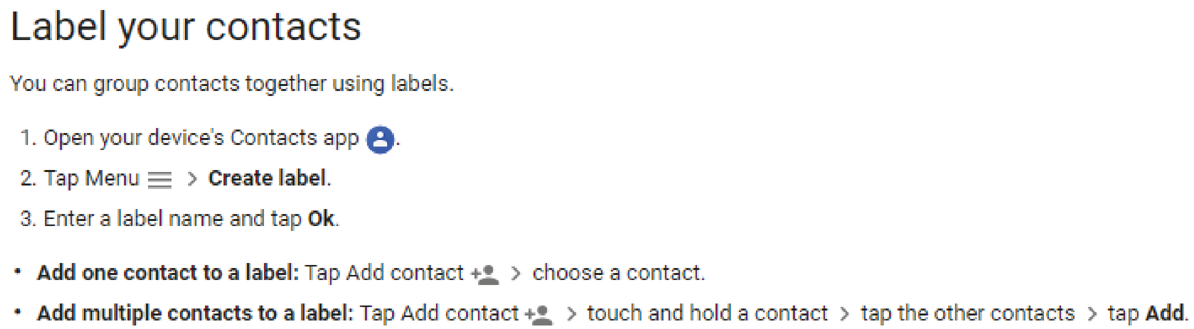
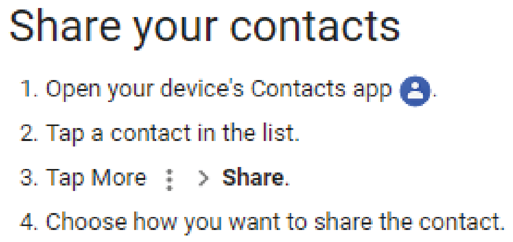
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 See your contacts 1. Open your device's Contacts app. 2. Tap Menu. - **See contacts by label:** Choose a label from the list. - **See contacts for another account:** Tap Down arrow > pick an account. - **See the contacts for all your accounts:** Choose **All contacts**. **Tip:** If you have multiple contacts with the same information, the information will be grouped into one contact. - **See your Google Account contacts on the web:** Go to Google Contacts. https://support.google.com/android/answer/6118731?hl=en&ref_topic=6118711  Label your contacts You can group contacts together using labels. 1. Open your device's Contacts app. 2. Tap Menu > **Create label**. 3. Enter a label name and tap **Ok**. - **Add one contact to a label:** Tap Add contact > choose a contact. - **Add multiple contacts to a label:** Tap Add contact > touch and hold a contact > tap the other contacts > tap **Add**. https://support.google.com/android/answer/6118731?hl=en&ref_topic=6118711  Share your contacts 1. Open your device's Contacts app. 2. Tap a contact in the list. 3. Tap More ⋮ > **Share**. 4. Choose how you want to share the contact. https://support.google.com/android/answer/6118731?hl=en&ref_topic=6118711

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

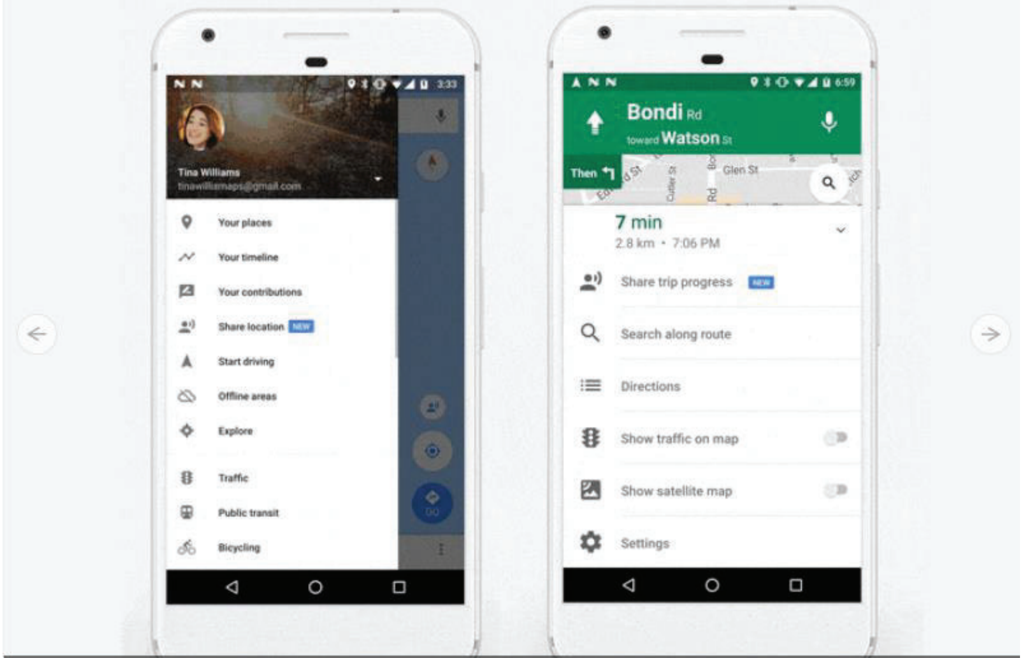
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Tina Williams tinawilliamsmaps@gmail.com Your places Your timeline Your contributions Share location NEW Start driving Offline areas Explore Traffic Public transit Bicycling 3:33 Bondi Rd toward Watson St Then Glen St 7 min 2.8 km • 7:06 PM Share trip progress NEW Search along route Directions Show traffic on map Show satellite map Settings 6:59 Ways to start location sharing: (left) from the app navigation drawer, and (right) from the menu, you can drag up while navigating somewhere. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

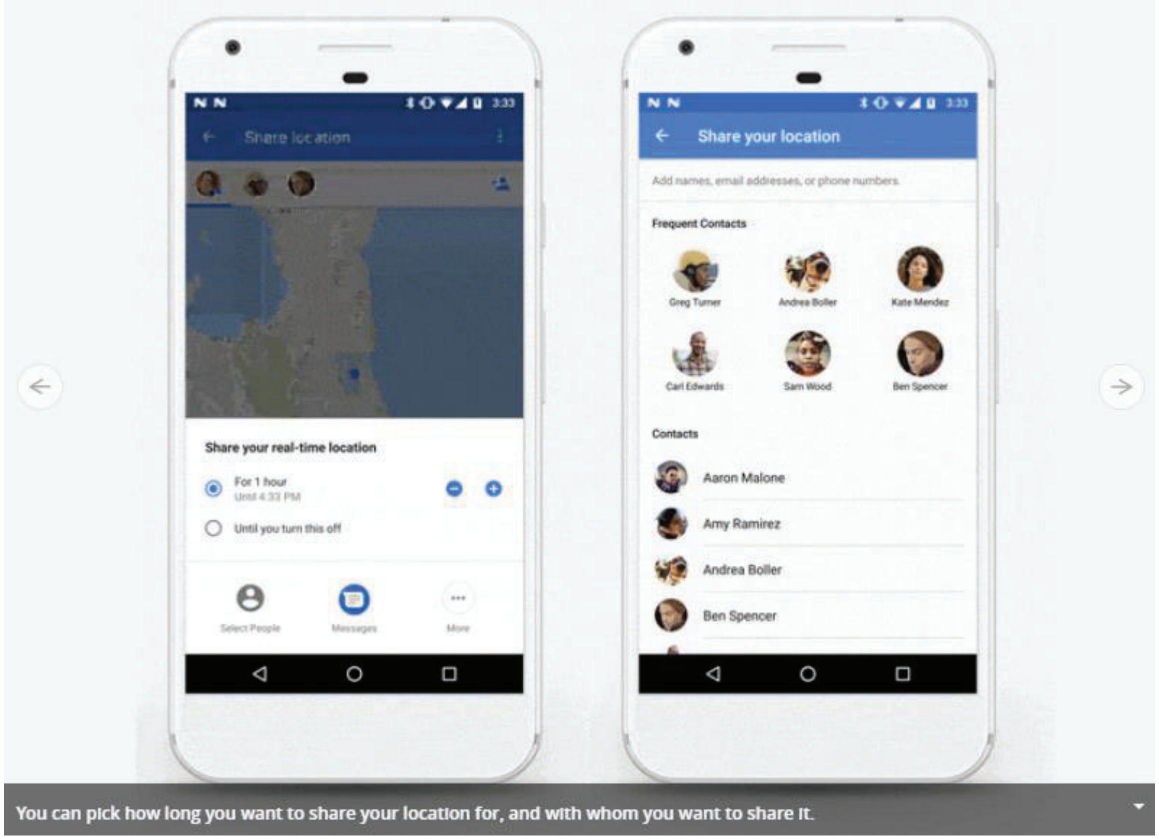
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Share location Share your real-time location For 1 hour Until 4:33 PM Until you turn this off Select People Messages More Share your location Add names, email addresses, or phone numbers Frequent Contacts Greg Turner Andrea Boller Kate Mendez Carl Edwards Sam Wood Ben Spencer Contacts Aaron Malone Amy Ramirez Andrea Boller Ben Spencer You can pick how long you want to share your location for, and with whom you want to share it. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

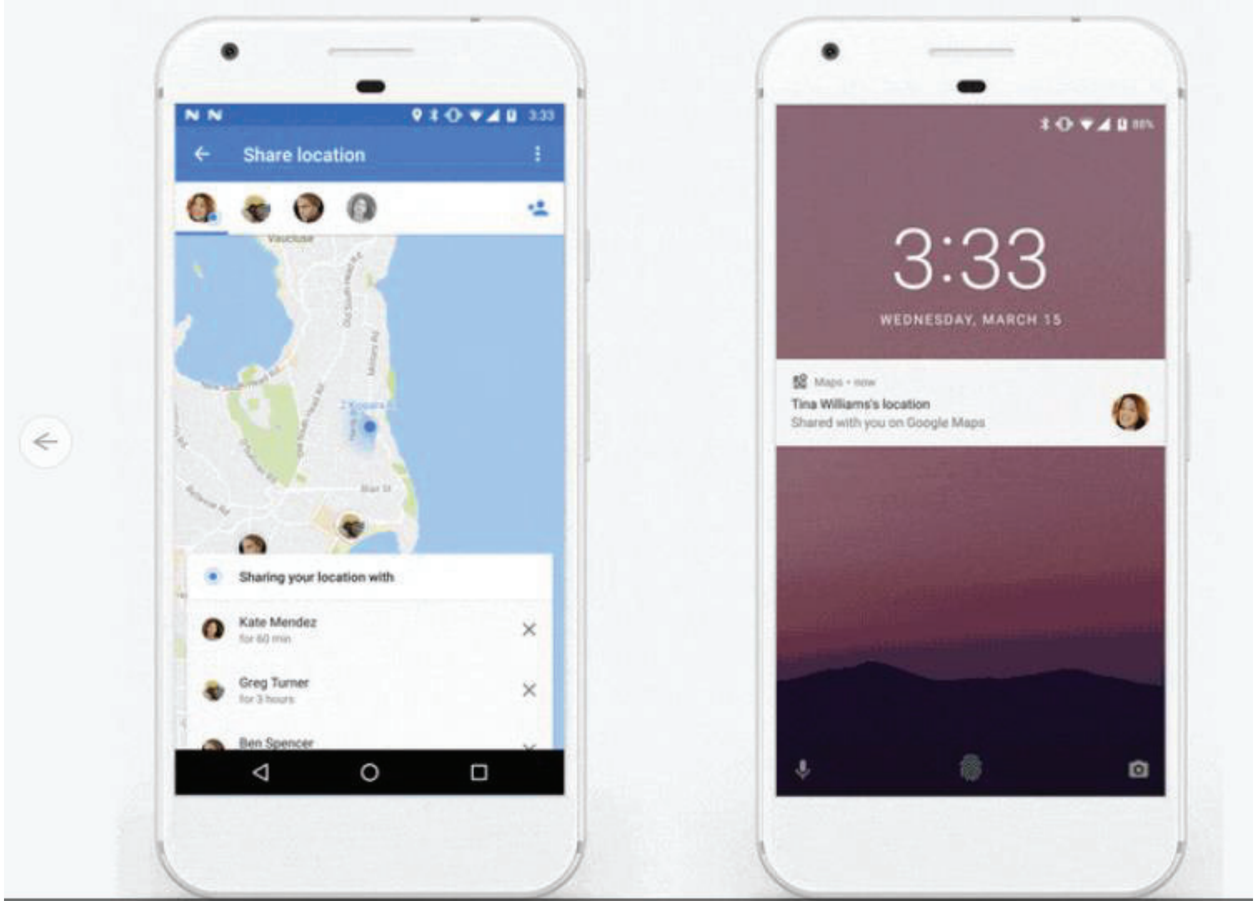
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Share location Sharing your location with Kate Mendez for 60 min Greg Turner for 3 hours Ben Spencer 3:33 WEDNESDAY, MARCH 15 Maps • now Tina Williams's location Shared with you on Google Maps People you've shared with will get a notification from Google Maps. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

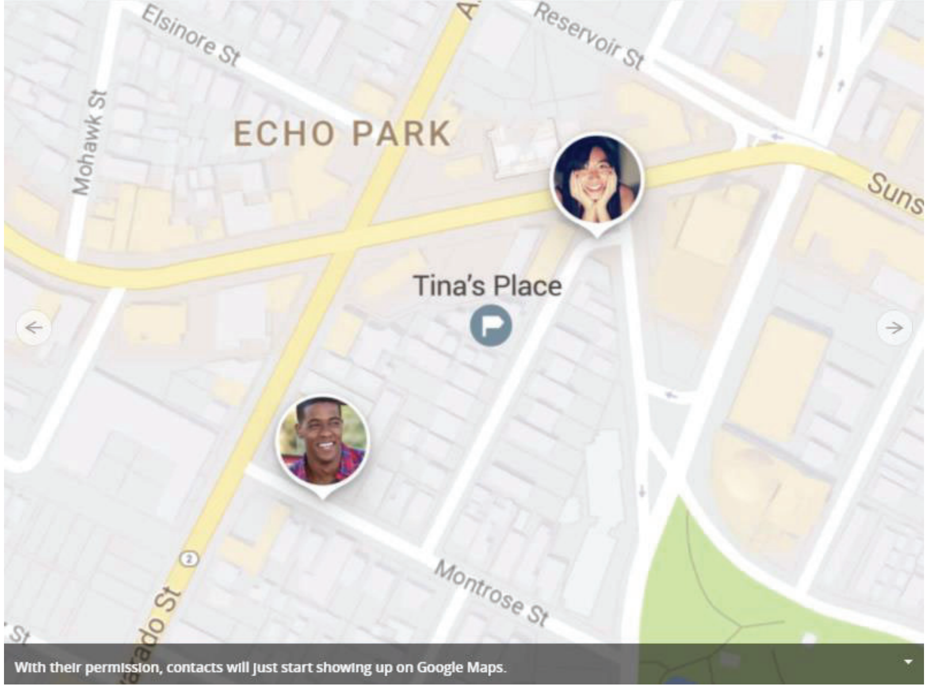
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Elsinore St Reservoir St Mohawk St ECHO PARK Suns Tina's Place Montrose St ado St With their permission, contacts will just start showing up on Google Maps. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/ **Exemplary Find My Device Screenshots**:

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

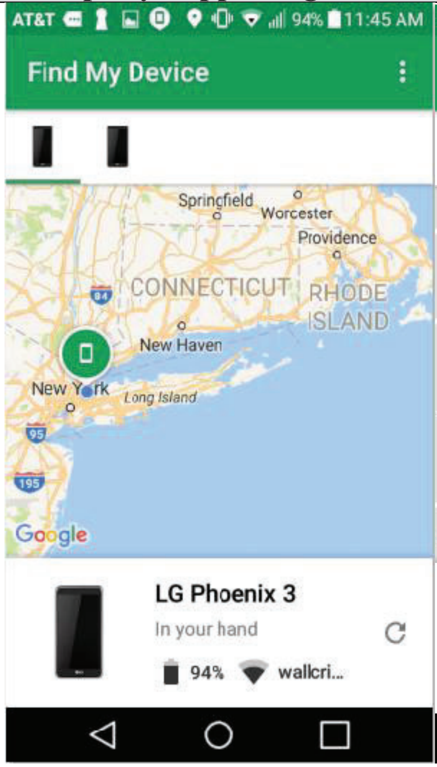
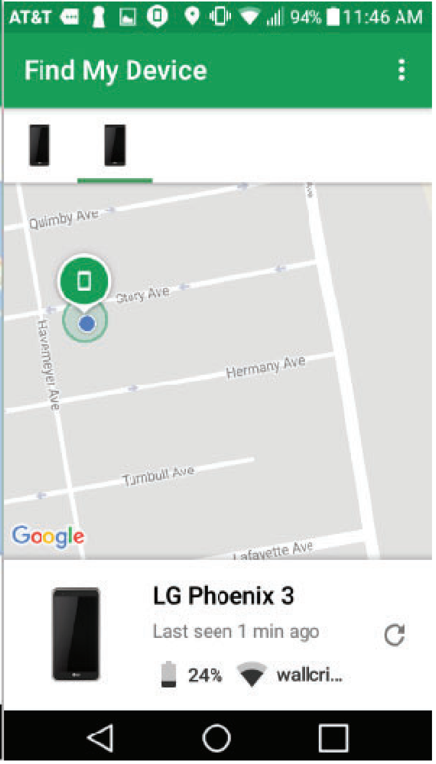
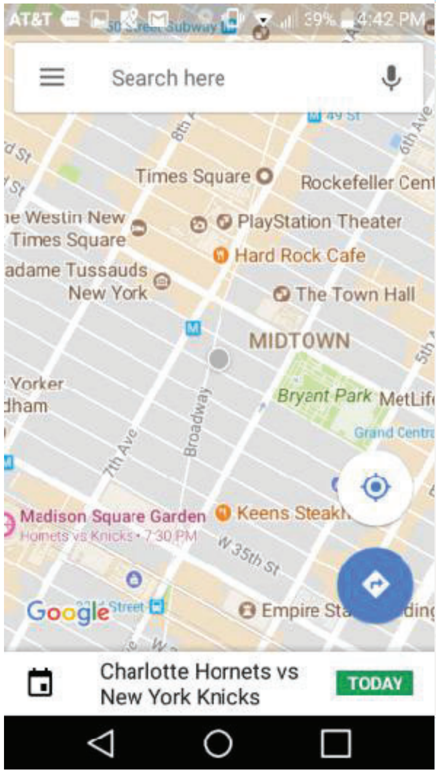
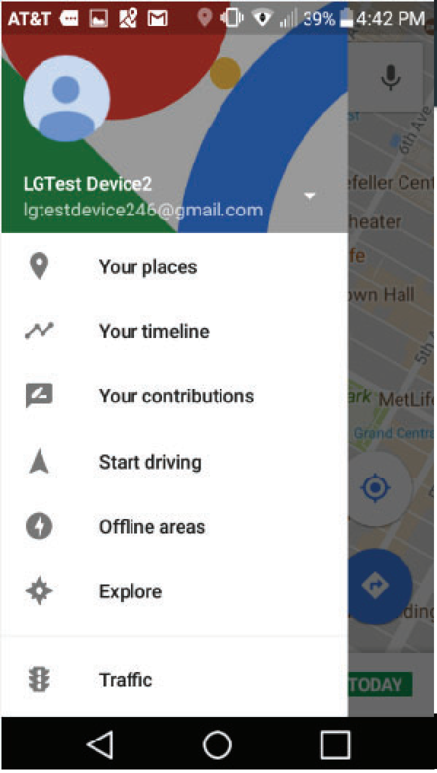
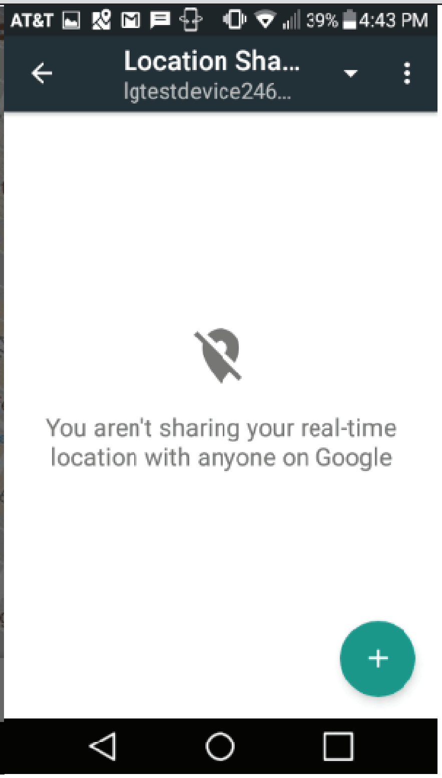
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 94% 11:45 AM Find My Device Springfield Worcester Providence CONNECTICUT RHODE ISLAND New Haven New York Long Island 84 95 195 Google LG Phoenix 3 In your hand 94% wallcri...  AT&T 94% 11:46 AM Find My Device Quimby Ave Story Ave Havemeyer Ave Hermany Ave Turnbull Ave Lafayette Ave Google LG Phoenix 3 Last seen 1 min ago 24% wallcri... **<u>Exemplary Google Maps Screenshots</u>:**

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 39% 4:42 PM Search here Times Square Rockefeller Cent Westin New Times Square PlayStation Theater Hard Rock Cafe Madame Tussauds New York The Town Hall MIDTOWN Bryant Park MetLife Grand Central Broadway 7th Ave Madison Square Garden Hornets vs Knicks • 7:30 PM Keens Steak W 35th St Empire State Building Google Charlotte Hornets vs New York Knicks TODAY  AT&T 39% 4:42 PM LGTest Device2 lg:estdevice246@gmail.com Your places Your timeline Your contributions Start driving Offline areas Explore Traffic  You aren't sharing your real-time location with anyone on Google

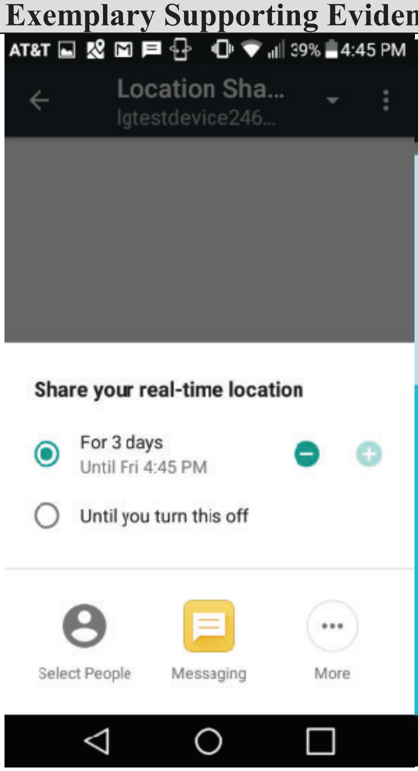
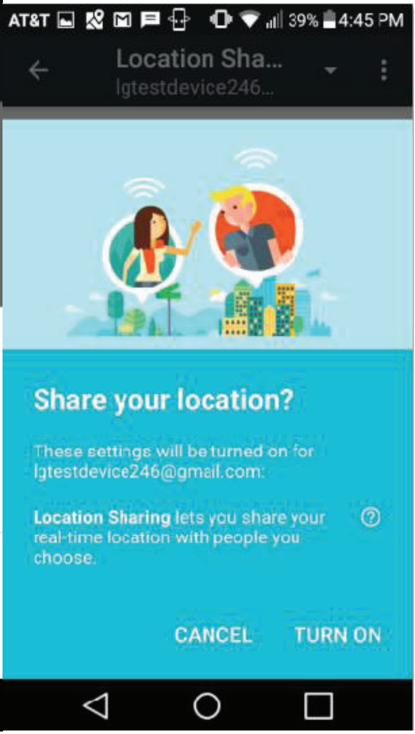
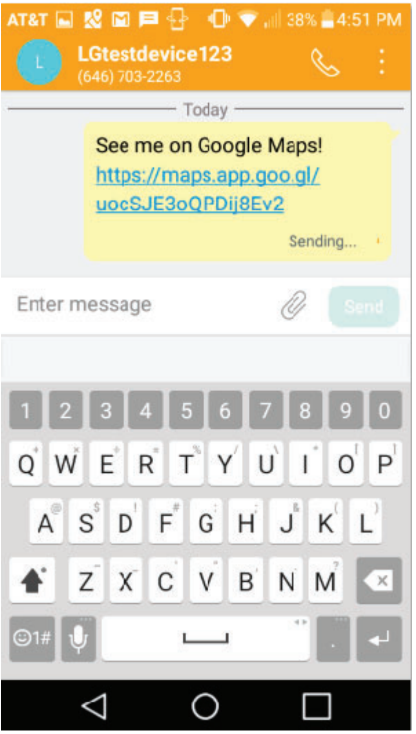
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Location Sha... lgtestdevice246... Share your real-time location ● For 3 days Until Fri 4:45 PM ○ Until you turn this off  Location Sha... lgtestdevice246... Share your location? These settings will be turned on for lgtestdevice246@gmail.com: Location Sharing lets you share your real-time location with people you choose.  LGtestdevice123 (646) 703-2263 Today See me on Google Maps! https://maps.app.goo.gl/uocSJE3oQPDij8Ev2 Sending... Enter message AT&T 38% 4:51 PM Send 1 2 3 4 5 6 7 8 9 0 Q W E R T Y U I O P A S D F G H J K L Z X C V B N M ☺1#

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

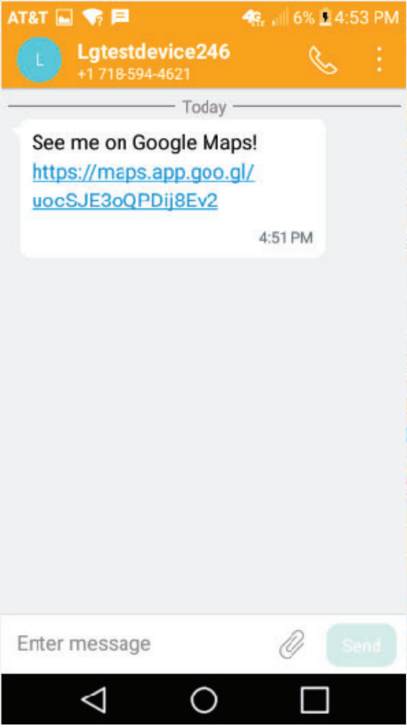
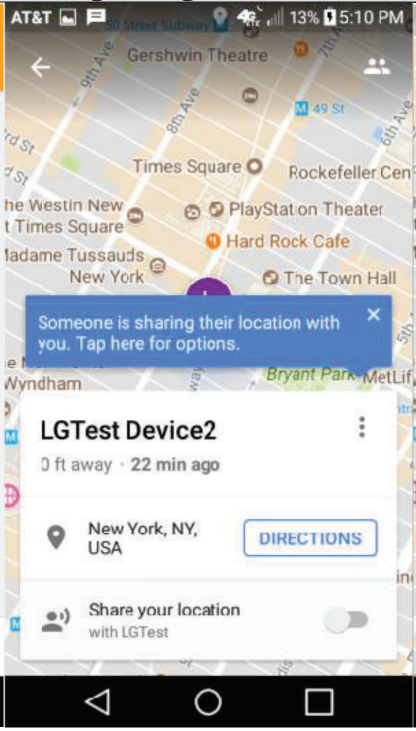
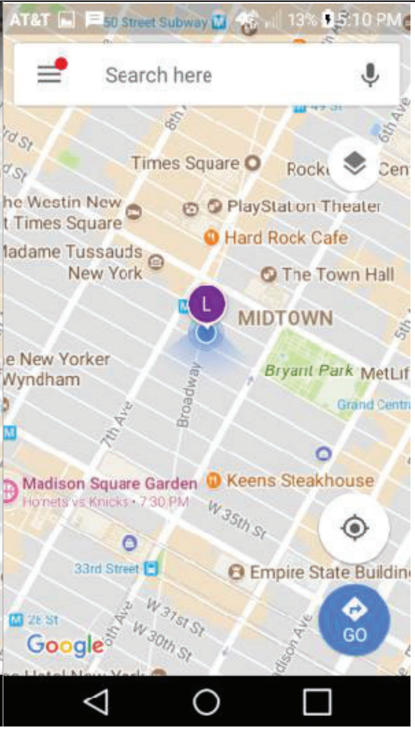
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 6% 4:53 PM Lgtestdevice246 +1 718-594-4621 Today See me on Google Maps! https://maps.app.goo.gl/uocSJE3oQPDij8Ev2 4:51 PM Enter message Send  AT&T 13% 5:10 PM Someone is sharing their location with you. Tap here for options. LGTest Device2 0 ft away · 22 min ago New York, NY, USA DIRECTIONS Share your location with LGTest  AT&T 13% 5:10 PM Search here Times Square PlayStation Theater Hard Rock Cafe The Town Hall MIDTOWN Bryant Park Madison Square Garden Keens Steakhouse Empire State Building GO Google

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

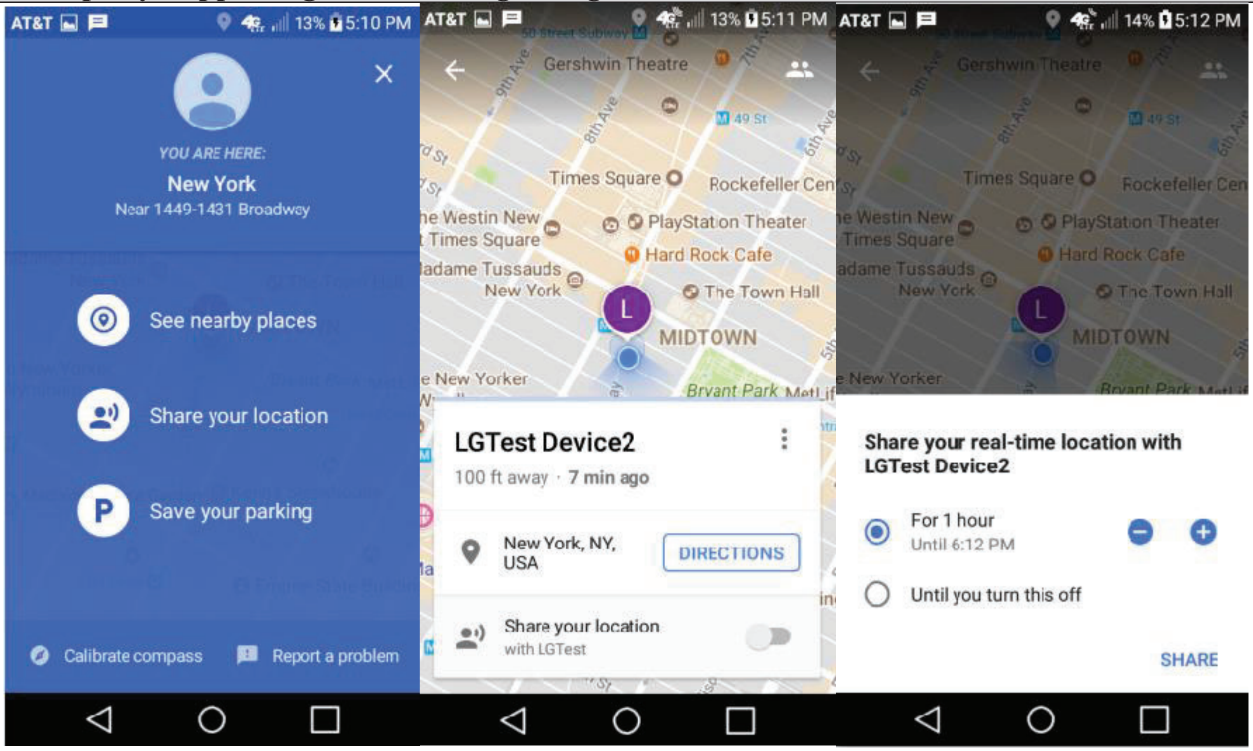
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 13% 5:10 PM YOU ARE HERE: New York Near 1449-1431 Broadway See nearby places Share your location Save your parking Calibrate compass Report a problem AT&T 13% 5:11 PM LGTest Device2 100 ft away · 7 min ago New York, NY, USA DIRECTIONS Share your location with LGTest AT&T 14% 5:12 PM Share your real-time location with LGTest Device2 ● For 1 hour Until 6:12 PM ○ Until you turn this off SHARE

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

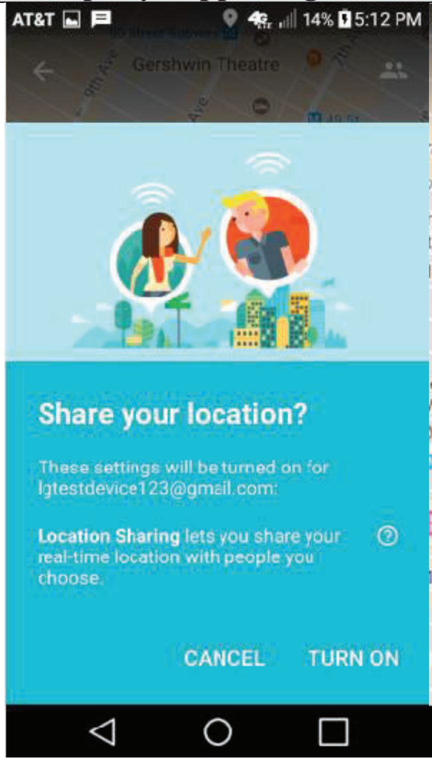
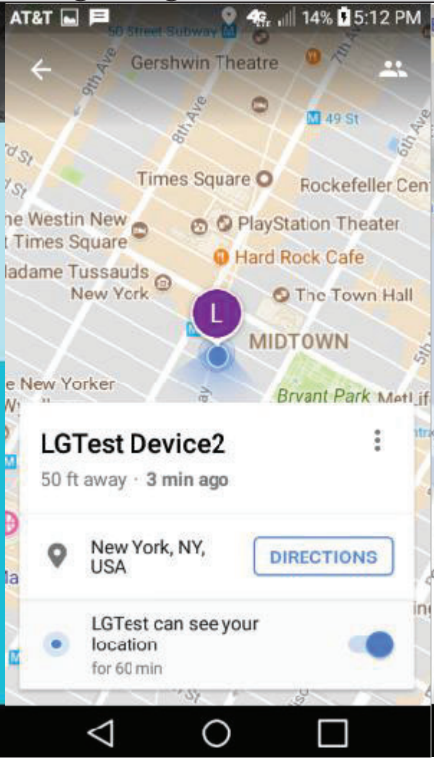
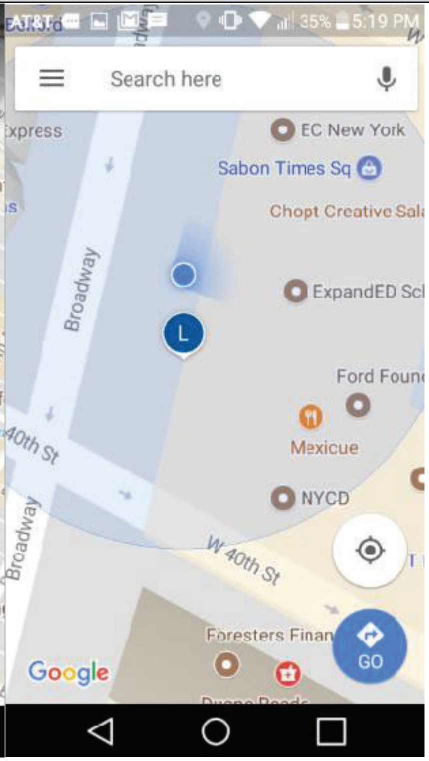
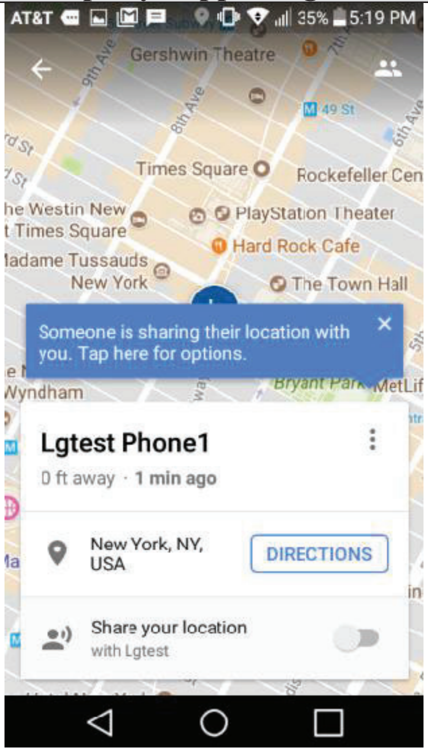
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 14% 5:12 PM Share your location? These settings will be turned on for lgtestdevice123@gmail.com: **Location Sharing** lets you share your real-time location with people you choose. CANCEL TURN ON  AT&T 14% 5:12 PM LGTest Device2 50 ft away · 3 min ago New York, NY, USA DIRECTIONS LGTest can see your location for 60 min Gershwin Theatre Times Square Rockefeller Cen PlayStation Theater Hard Rock Cafe The Town Hall MIDTOWN Bryant Park L  AT&T 35% 5:19 PM Search here EC New York Sabon Times Sq Chopt Creative Sala ExpandED Sc Ford Foun Mexicue NYCD Broadway W 40th St Foresters Finan GO Google L

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Someone is sharing their location with you. Tap here for options. Lgtest Phone1 0 ft away · 1 min ago New York, NY, USA Share your location with Lgtest

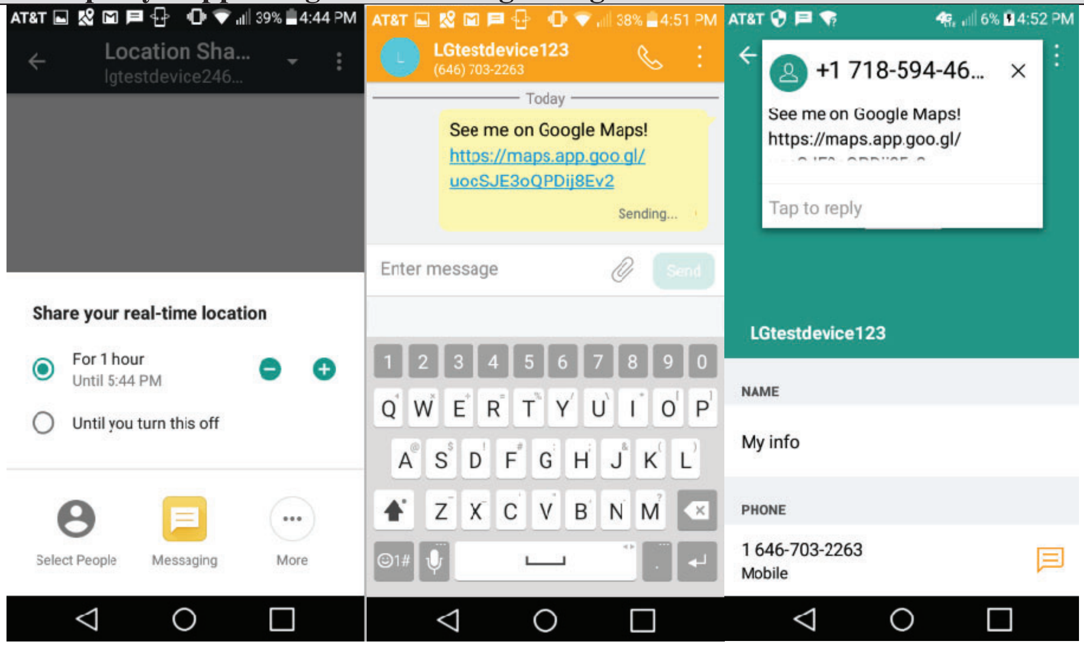
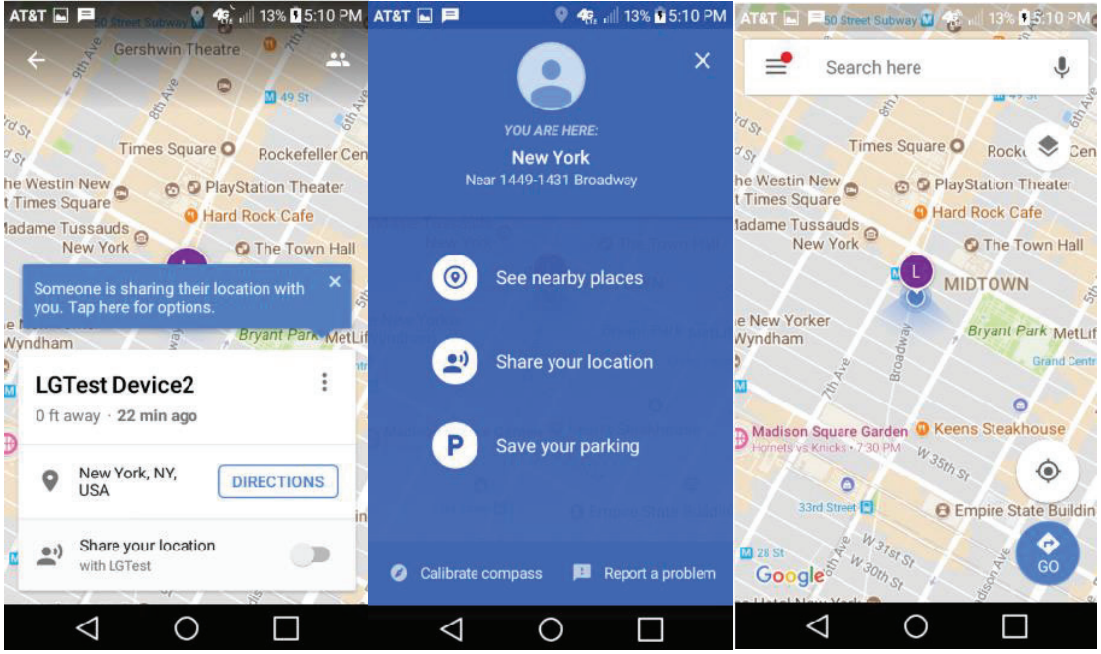
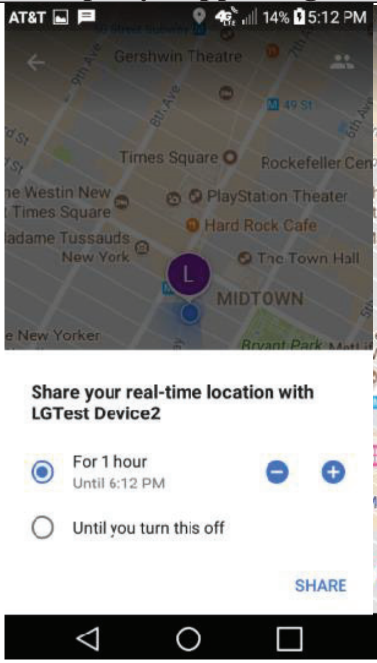
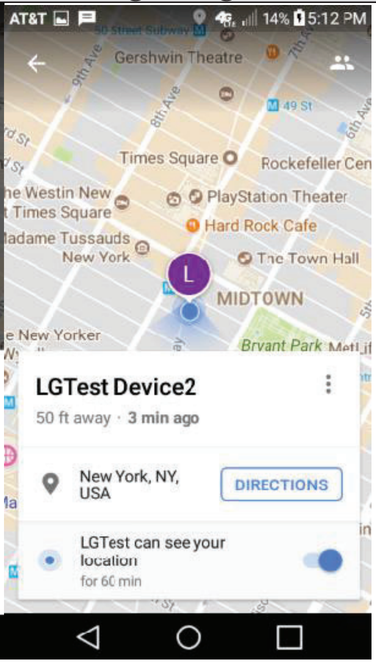
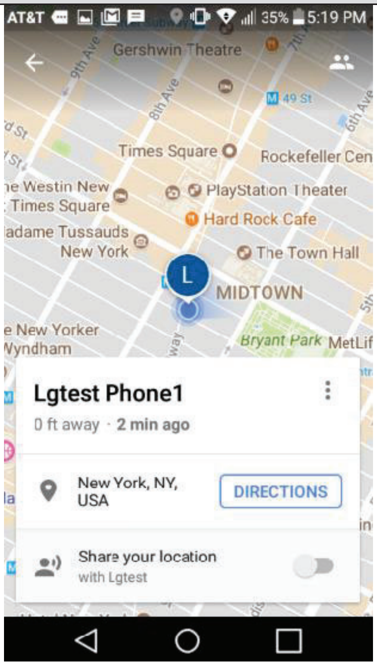
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Location Sha... lgtestdevice246... Share your real-time location ○ For 1 hour Until 5:44 PM ○ Until you turn this off LGtestdevice123 (646) 703-2263 Today See me on Google Maps! https://maps.app.goo.gl/uocSJE3oQPDij8Ev2 Sending... +1 718-594-46... See me on Google Maps! https://maps.app.goo.gl/ Tap to reply LGtestdevice123 NAME My info PHONE 1 646-703-2263 Mobile AT&T 39% 4:44 PM Select People Messaging More AT&T 38% 4:51 PM Enter message Send AT&T 6% 4:52 PM

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Someone is sharing their location with you. Tap here for options. LGTest Device2 0 ft away · 22 min ago New York, NY, USA DIRECTIONS Share your location with LGTest YOU ARE HERE: New York Near 1449-1431 Broadway See nearby places Share your location Save your parking Calibrate compass Report a problem Search here GO

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 14% 5:12 PM Gershwin Theatre Times Square Rockefeller Cen PlayStation Theater Hard Rock Cafe The Town Hall MIDTOWN L Share your real-time location with LGTest Device2 ○ For 1 hour Until 6:12 PM ○ Until you turn this off SHARE  AT&T 14% 5:12 PM Gershwin Theatre Times Square Rockefeller Cen PlayStation Theater Hard Rock Cafe The Town Hall MIDTOWN L LGTest Device2 50 ft away · 3 min ago New York, NY, USA DIRECTIONS LGTest can see your location for 60 min  AT&T 35% 5:19 PM Gershwin Theatre Times Square Rockefeller Cen PlayStation Theater Hard Rock Cafe The Town Hall MIDTOWN L Lgtest Phone1 0 ft away · 2 min ago New York, NY, USA DIRECTIONS Share your location with Lgtest **Exemplary Source Code**: The above functionality is performed at least in part by the following publicly available source code and/or source code that invokes or is invoked by the following source code (or a substantially similar copy compiled and loaded onto the Accused Products by ZTE). AGIS reserves the right to amend these contentions to include additional source code as discovery progresses and as additional source code is made available.

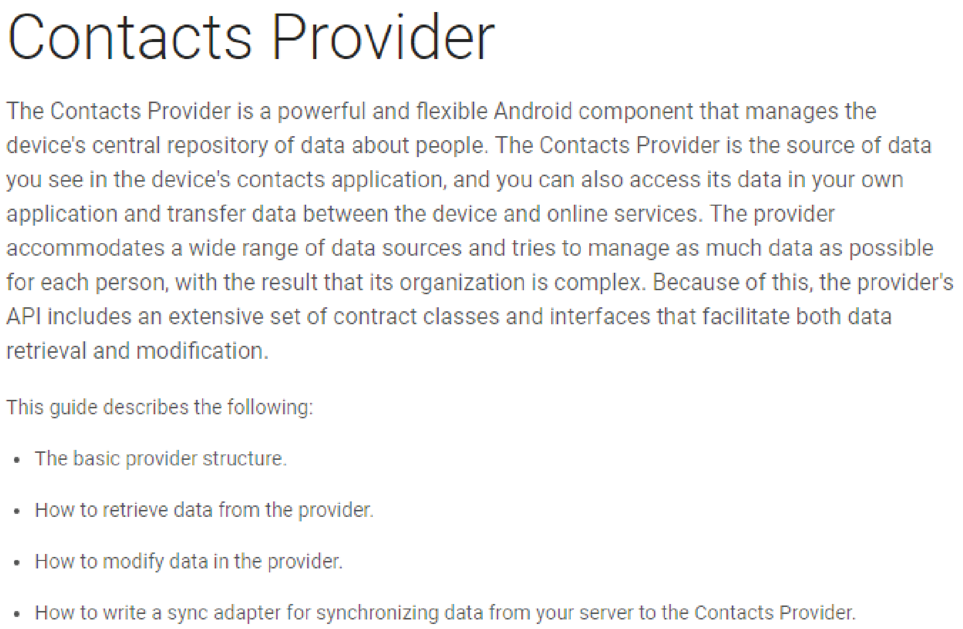
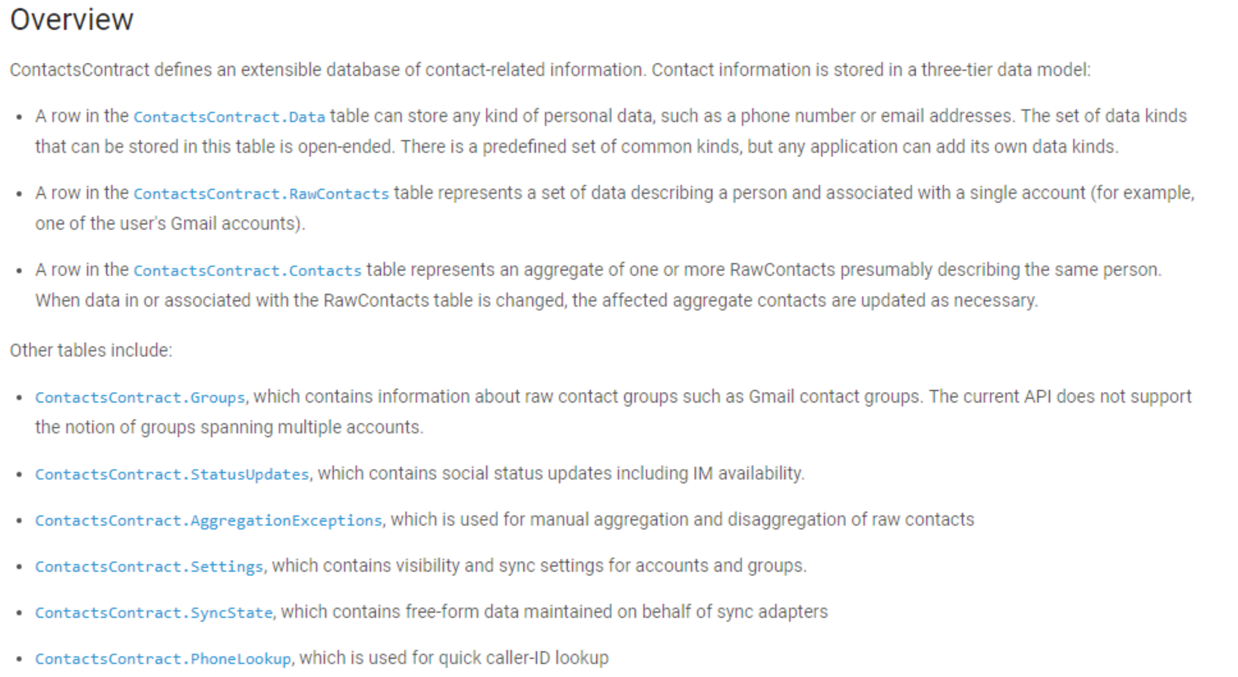
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Contacts Provider The Contacts Provider is a powerful and flexible Android component that manages the device's central repository of data about people. The Contacts Provider is the source of data you see in the device's contacts application, and you can also access its data in your own application and transfer data between the device and online services. The provider accommodates a wide range of data sources and tries to manage as much data as possible for each person, with the result that its organization is complex. Because of this, the provider's API includes an extensive set of contract classes and interfaces that facilitate both data retrieval and modification. This guide describes the following: • The basic provider structure. • How to retrieve data from the provider. • How to modify data in the provider. • How to write a sync adapter for synchronizing data from your server to the Contacts Provider. https://developer.android.com/guide/topics/providers/contacts-provider.html

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>


Overview

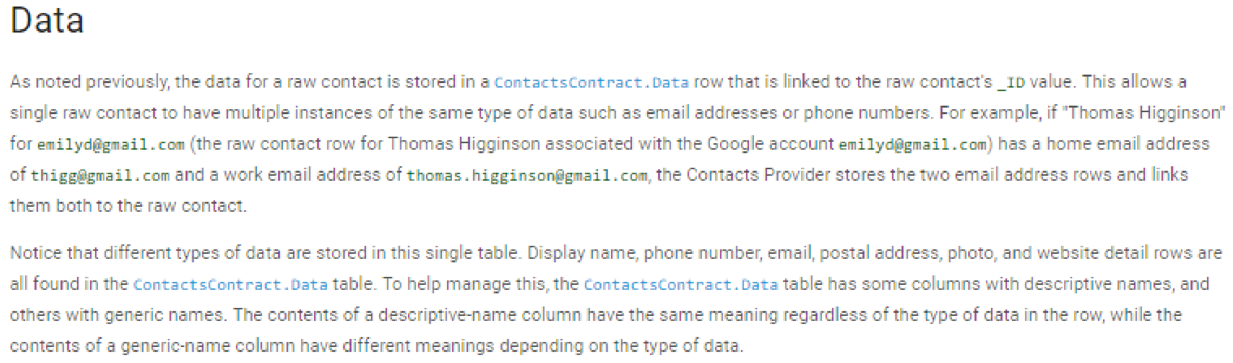
ContactsContract defines an extensible database of contact-related information. Contact information is stored in a three-tier data model:

- A row in the `ContactsContract.Data` table can store any kind of personal data, such as a phone number or email addresses. The set of data kinds that can be stored in this table is open-ended. There is a predefined set of common kinds, but any application can add its own data kinds.
- A row in the `ContactsContract.RawContacts` table represents a set of data describing a person and associated with a single account (for example, one of the user's Gmail accounts).
- A row in the `ContactsContract.Contacts` table represents an aggregate of one or more RawContacts presumably describing the same person. When data in or associated with the RawContacts table is changed, the affected aggregate contacts are updated as necessary.

Other tables include:

- `ContactsContract.Groups`, which contains information about raw contact groups such as Gmail contact groups. The current API does not support the notion of groups spanning multiple accounts.
- `ContactsContract.StatusUpdates`, which contains social status updates including IM availability.
- `ContactsContract.AggregationExceptions`, which is used for manual aggregation and disaggregation of raw contacts
- `ContactsContract.Settings`, which contains visibility and sync settings for accounts and groups.
- `ContactsContract.SyncState`, which contains free-form data maintained on behalf of sync adapters
- `ContactsContract.PhoneLookup`, which is used for quick caller-ID lookup

https://developer.android.com/reference/android/provider/ContactsContract.html

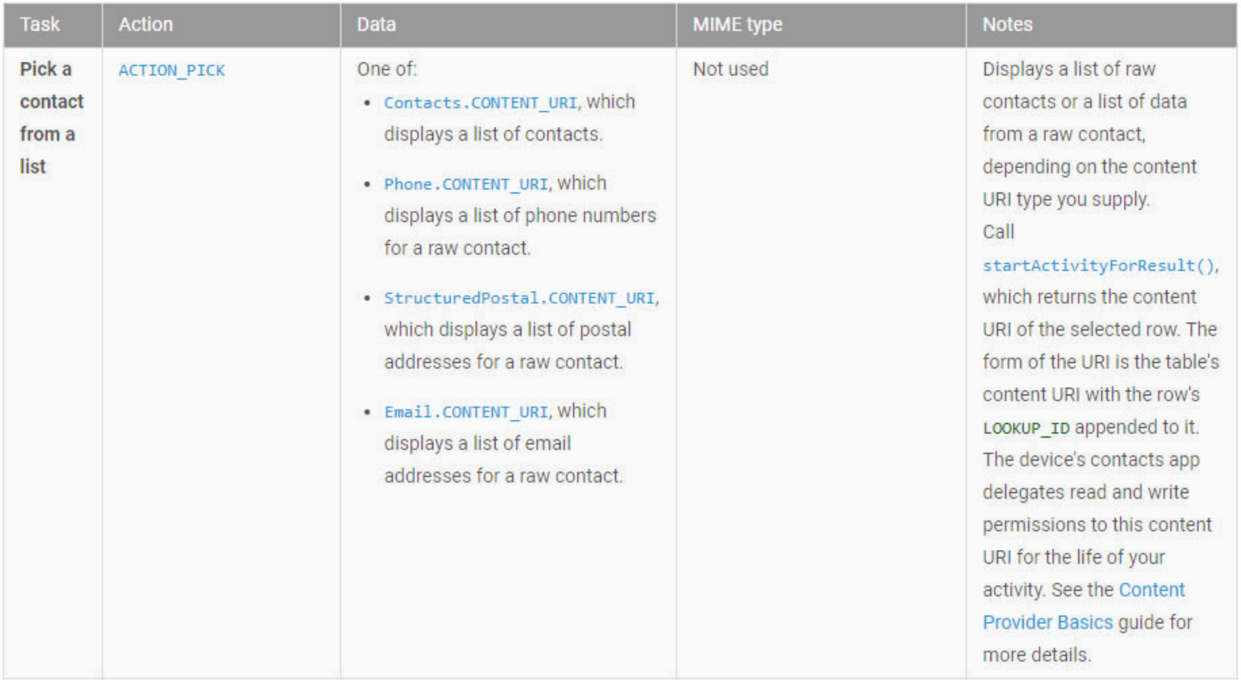


Data

As noted previously, the data for a raw contact is stored in a `ContactsContract.Data` row that is linked to the raw contact's `_ID` value. This allows a single raw contact to have multiple instances of the same type of data such as email addresses or phone numbers. For example, if "Thomas Higginson" for `emilyd@gmail.com` (the raw contact row for Thomas Higginson associated with the Google account `emilyd@gmail.com`) has a home email address of `thigg@gmail.com` and a work email address of `thomas.higginson@gmail.com`, the Contacts Provider stores the two email address rows and links them both to the raw contact.

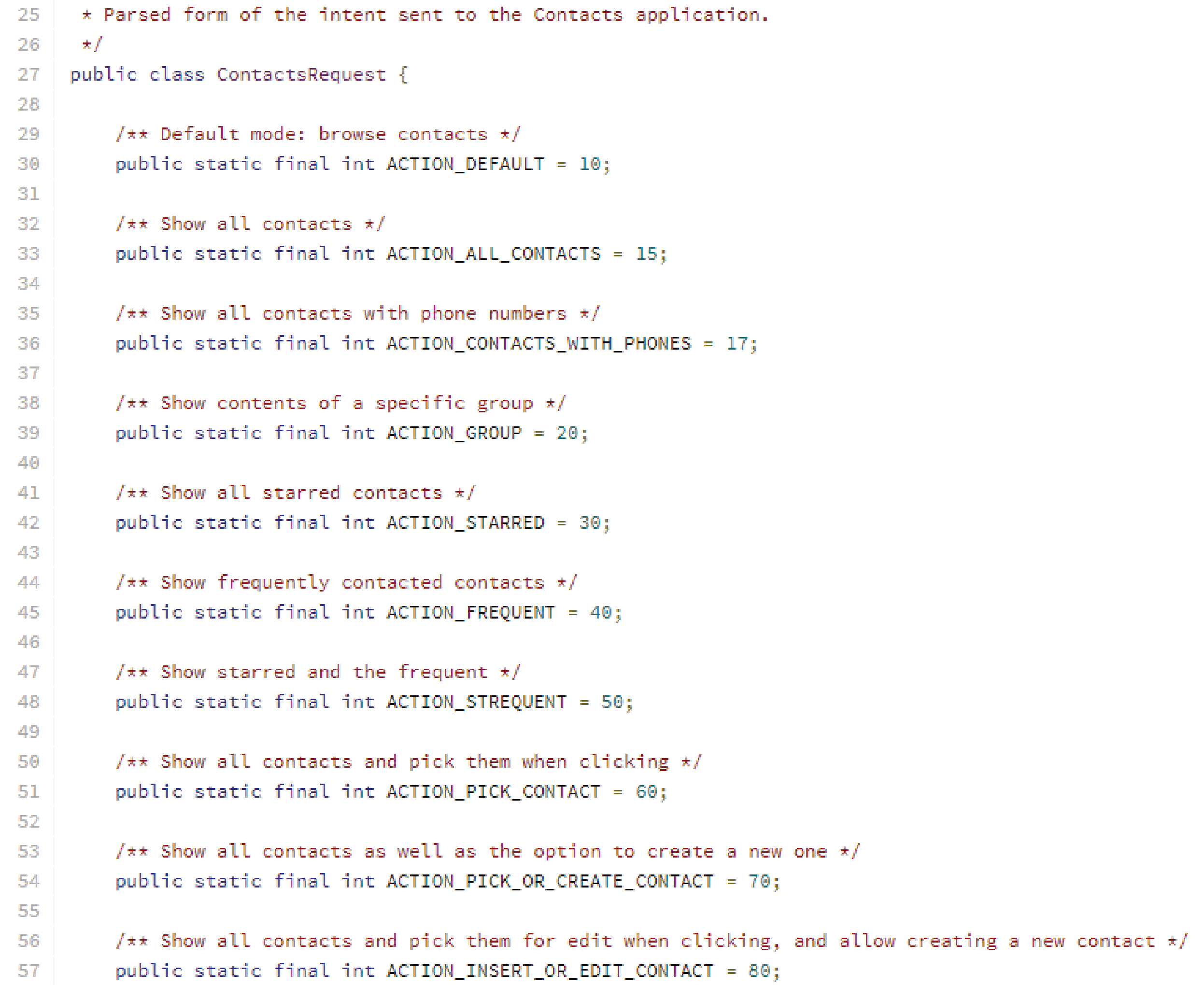
Notice that different types of data are stored in this single table. Display name, phone number, email, postal address, photo, and website detail rows are all found in the `ContactsContract.Data` table. To help manage this, the `ContactsContract.Data` table has some columns with descriptive names, and others with generic names. The contents of a descriptive-name column have the same meaning regardless of the type of data in the row, while the contents of a generic-name column have different meanings depending on the type of data.

https://developer.android.com/guide/topics/providers/contacts-provider.html
</td></tr>
</table>

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 https://developer.android.com/guide/topics/providers/contacts-provider.html

Task	Action	Data	MIME type	Notes
Pick a contact from a list	ACTION_PICK	One of: • Contacts.CONTENT_URI, which displays a list of contacts. • Phone.CONTENT_URI, which displays a list of phone numbers for a raw contact. • StructuredPostal.CONTENT_URI, which displays a list of postal addresses for a raw contact. • Email.CONTENT_URI, which displays a list of email addresses for a raw contact.	Not used	Displays a list of raw contacts or a list of data from a raw contact, depending on the content URI type you supply. Call startActivityForResult(), which returns the content URI of the selected row. The form of the URI is the table's content URI with the row's LOOKUP_ID appended to it. The device's contacts app delegates read and write permissions to this content URI for the life of your activity. See the Content Provider Basics guide for more details.

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 https://android.googlesource.com/platform/packages/apps/Contacts/+/nougat-mr1-release/src/com/android/contacts/list/ContactsRequest.java

```
 * Parsed form of the intent sent to the Contacts application.
 */
public class ContactsRequest {

    /** Default mode: browse contacts */
    public static final int ACTION_DEFAULT = 10;

    /** Show all contacts */
    public static final int ACTION_ALL_CONTACTS = 15;

    /** Show all contacts with phone numbers */
    public static final int ACTION_CONTACTS_WITH_PHONES = 17;

    /** Show contents of a specific group */
    public static final int ACTION_GROUP = 20;

    /** Show all starred contacts */
    public static final int ACTION_STARRED = 30;

    /** Show frequently contacted contacts */
    public static final int ACTION_FREQUENT = 40;

    /** Show starred and the frequent */
    public static final int ACTION_STREQUENT = 50;

    /** Show all contacts and pick them when clicking */
    public static final int ACTION_PICK_CONTACT = 60;

    /** Show all contacts as well as the option to create a new one */
    public static final int ACTION_PICK_OR_CREATE_CONTACT = 70;

    /** Show all contacts and pick them for edit when clicking, and allow creating a new contact */
    public static final int ACTION_INSERT_OR_EDIT_CONTACT = 80;
```

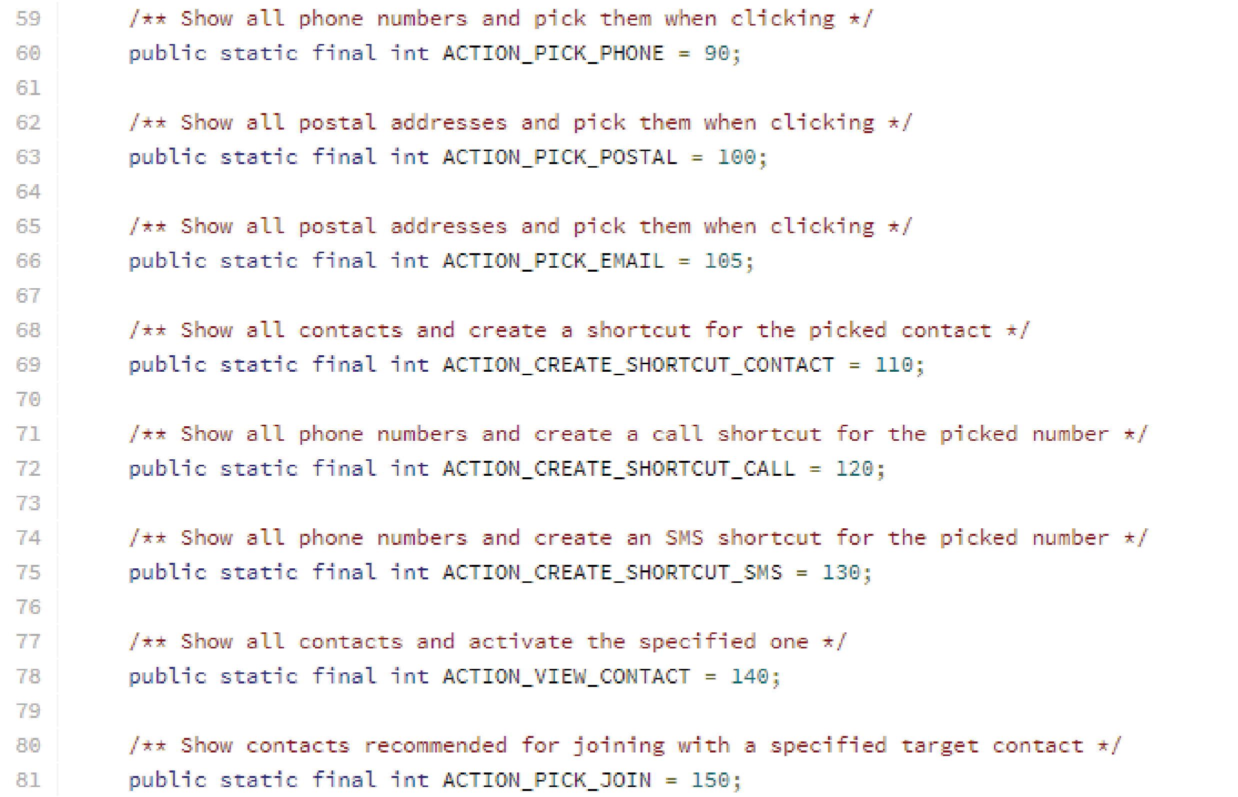
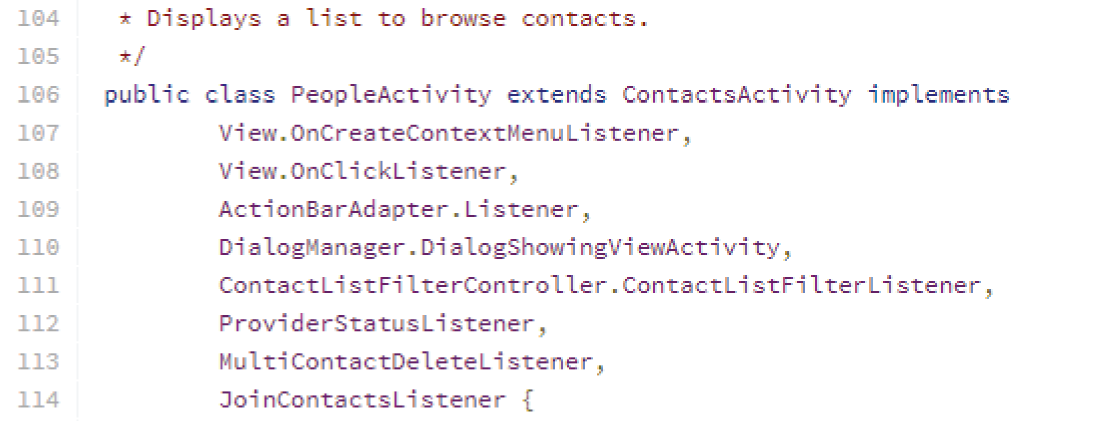
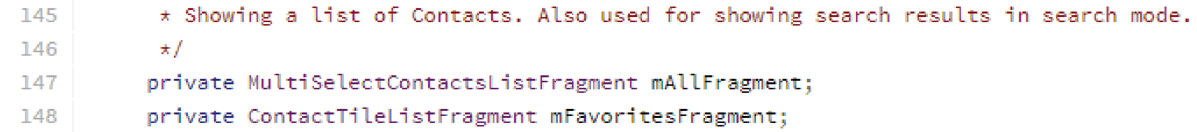
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 /** Show all phone numbers and pick them when clicking */ public static final int ACTION_PICK_PHONE = 90; /** Show all postal addresses and pick them when clicking */ public static final int ACTION_PICK_POSTAL = 100; /** Show all postal addresses and pick them when clicking */ public static final int ACTION_PICK_EMAIL = 105; /** Show all contacts and create a shortcut for the picked contact */ public static final int ACTION_CREATE_SHORTCUT_CONTACT = 110; /** Show all phone numbers and create a call shortcut for the picked number */ public static final int ACTION_CREATE_SHORTCUT_CALL = 120; /** Show all phone numbers and create an SMS shortcut for the picked number */ public static final int ACTION_CREATE_SHORTCUT_SMS = 130; /** Show all contacts and activate the specified one */ public static final int ACTION_VIEW_CONTACT = 140; /** Show contacts recommended for joining with a specified target contact */ public static final int ACTION_PICK_JOIN = 150; https://android.googlesource.com/platform/packages/apps/Contacts/+/nougat-mr1-release/src/com/android/contacts/list/ContactsRequest.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>


```
 * Displays a list to browse contacts.
 */
public class PeopleActivity extends ContactsActivity implements
        View.OnCreateContextMenuListener,
        View.OnClickListener,
        ActionBarAdapter.Listener,
        DialogManager.DialogShowingViewActivity,
        ContactListFilterController.ContactListFilterListener,
        ProviderStatusListener,
        MultiContactDeleteListener,
        JoinContactsListener {
```

https://android.googlesource.com/platform/packages/apps/Contacts/+/nougat-mr1-release/src/com/android/contacts/activities/PeopleActivity.java

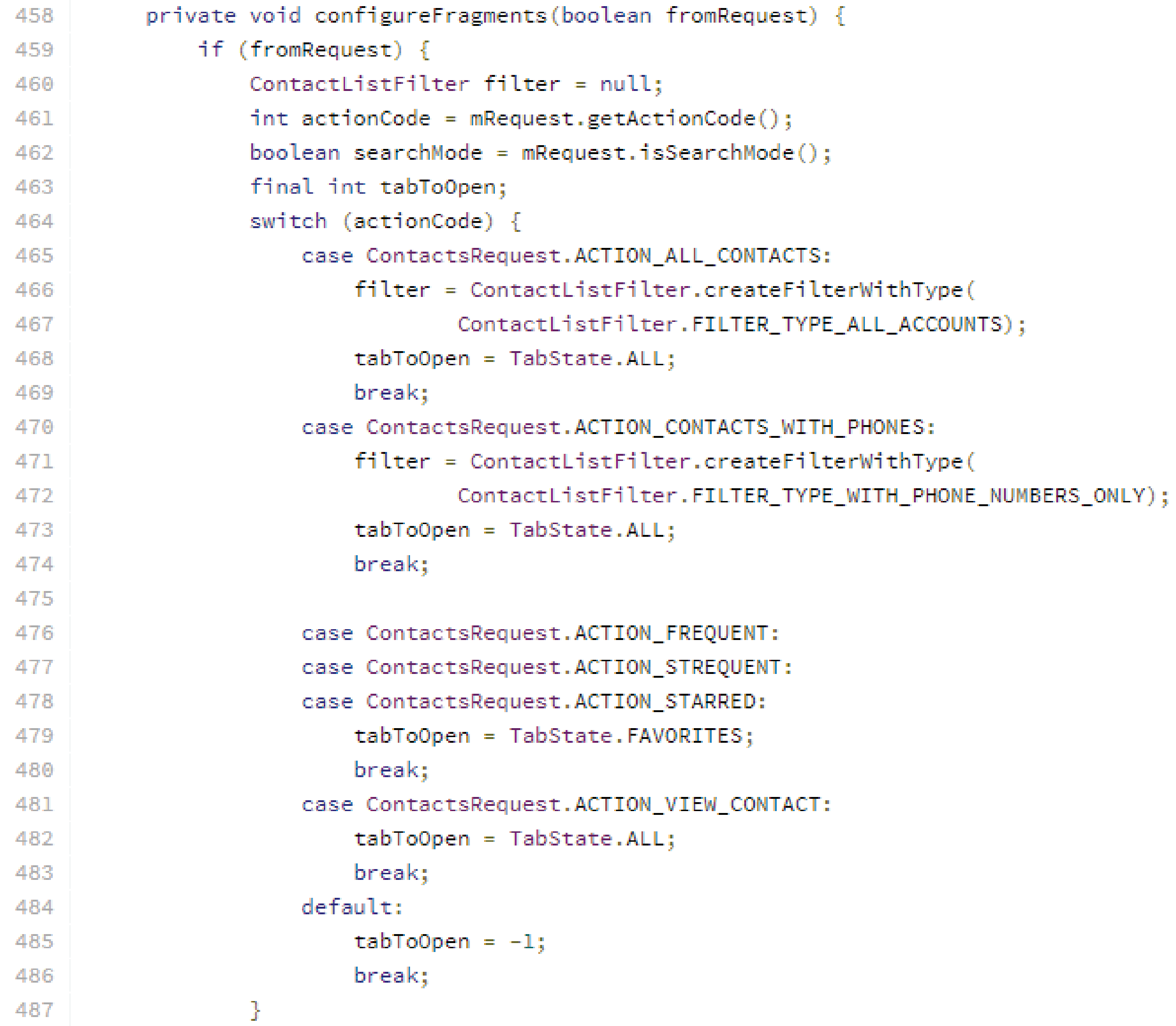


```
     * Showing a list of Contacts. Also used for showing search results in search mode.
     */
    private MultiSelectContactsListFragment mAllFragment;
    private ContactTileListFragment mFavoritesFragment;
```

https://android.googlesource.com/platform/packages/apps/Contacts/+/nougat-mr1-release/src/com/android/contacts/activities/PeopleActivity.java
</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products

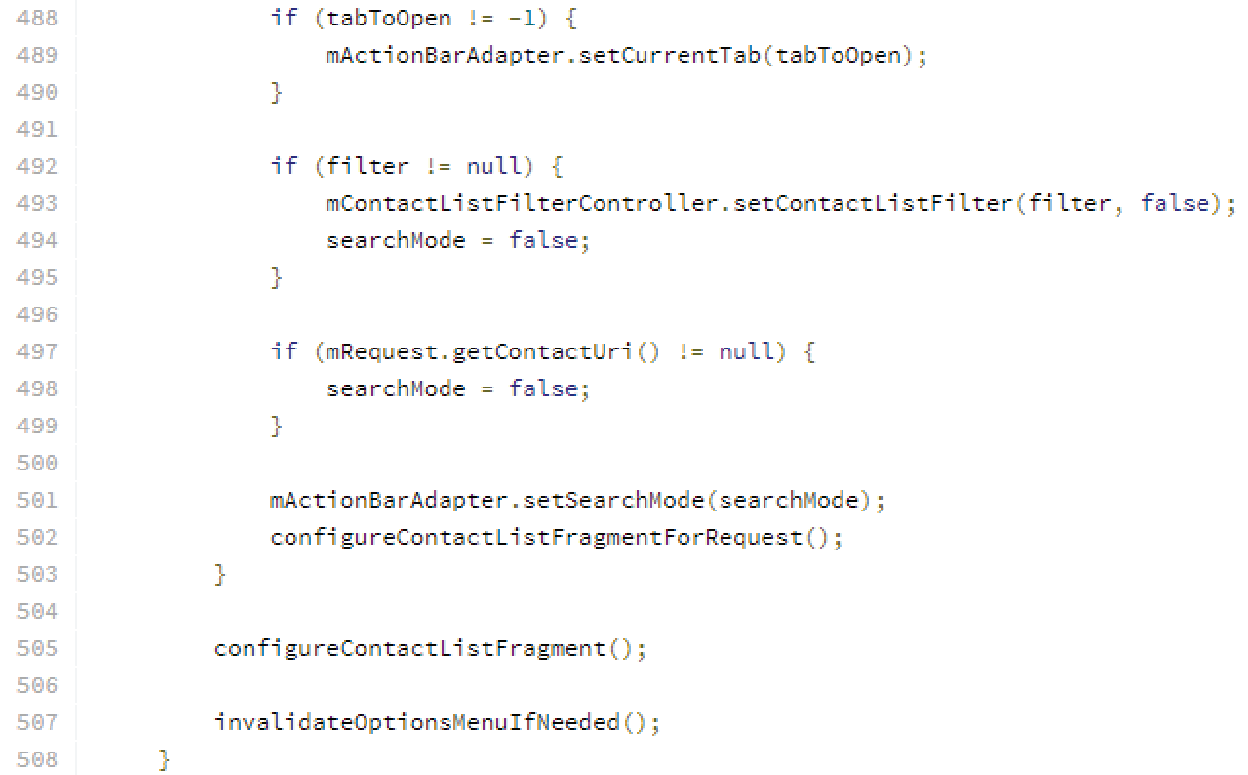


```
    private void configureFragments(boolean fromRequest) {
        if (fromRequest) {
            ContactListFilter filter = null;
            int actionCode = mRequest.getActionCode();
            boolean searchMode = mRequest.isSearchMode();
            final int tabToOpen;
            switch (actionCode) {
                case ContactsRequest.ACTION_ALL_CONTACTS:
                    filter = ContactListFilter.createFilterWithType(
                            ContactListFilter.FILTER_TYPE_ALL_ACCOUNTS);
                    tabToOpen = TabState.ALL;
                    break;
                case ContactsRequest.ACTION_CONTACTS_WITH_PHONES:
                    filter = ContactListFilter.createFilterWithType(
                            ContactListFilter.FILTER_TYPE_WITH_PHONE_NUMBERS_ONLY);
                    tabToOpen = TabState.ALL;
                    break;

                case ContactsRequest.ACTION_FREQUENT:
                case ContactsRequest.ACTION_STREQUENT:
                case ContactsRequest.ACTION_STARRED:
                    tabToOpen = TabState.FAVORITES;
                    break;
                case ContactsRequest.ACTION_VIEW_CONTACT:
                    tabToOpen = TabState.ALL;
                    break;
                default:
                    tabToOpen = -1;
                    break;
            }
```

https://android.googlesource.com/platform/packages/apps/Contacts/+/nougat-mr1-release/src/com/android/contacts/activities/PeopleActivity.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products

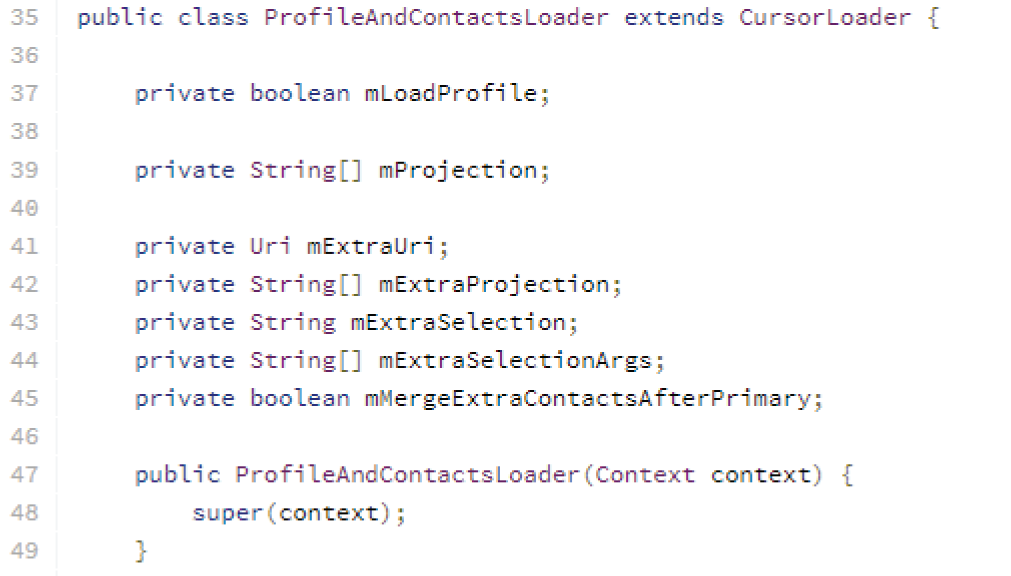


```
            if (tabToOpen != -1) {
                mActionBarAdapter.setCurrentTab(tabToOpen);
            }

            if (filter != null) {
                mContactListFilterController.setContactListFilter(filter, false);
                searchMode = false;
            }

            if (mRequest.getContactUri() != null) {
                searchMode = false;
            }

            mActionBarAdapter.setSearchMode(searchMode);
            configureContactListFragmentForRequest();
        }

        configureContactListFragment();

        invalidateOptionsMenuIfNeeded();
    }
```

https://android.googlesource.com/platform/packages/apps/Contacts/+/nougat-mr1-release/src/com/android/contacts/activities/PeopleActivity.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>



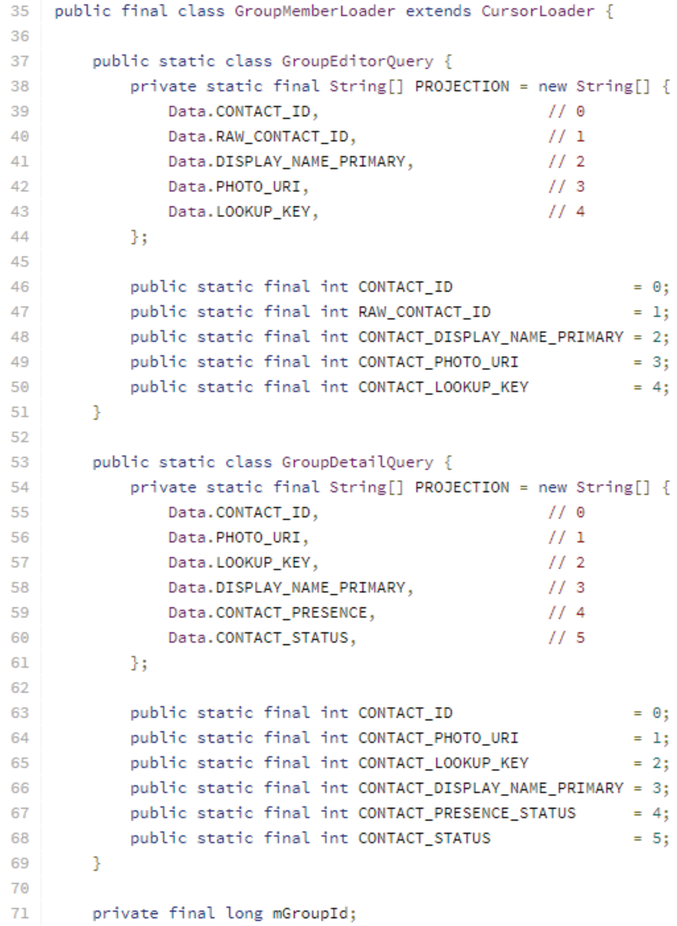
```
public class ProfileAndContactsLoader extends CursorLoader {

    private boolean mLoadProfile;

    private String[] mProjection;

    private Uri mExtraUri;
    private String[] mExtraProjection;
    private String mExtraSelection;
    private String[] mExtraSelectionArgs;
    private boolean mMergeExtraContactsAfterPrimary;

    public ProfileAndContactsLoader(Context context) {
        super(context);
    }
```

https://android.googlesource.com/platform/packages/apps/ContactsCommon/+/nougat-mr1-release/src/com/android/contacts/common/list/ProfileAndContactsLoader.java

</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products

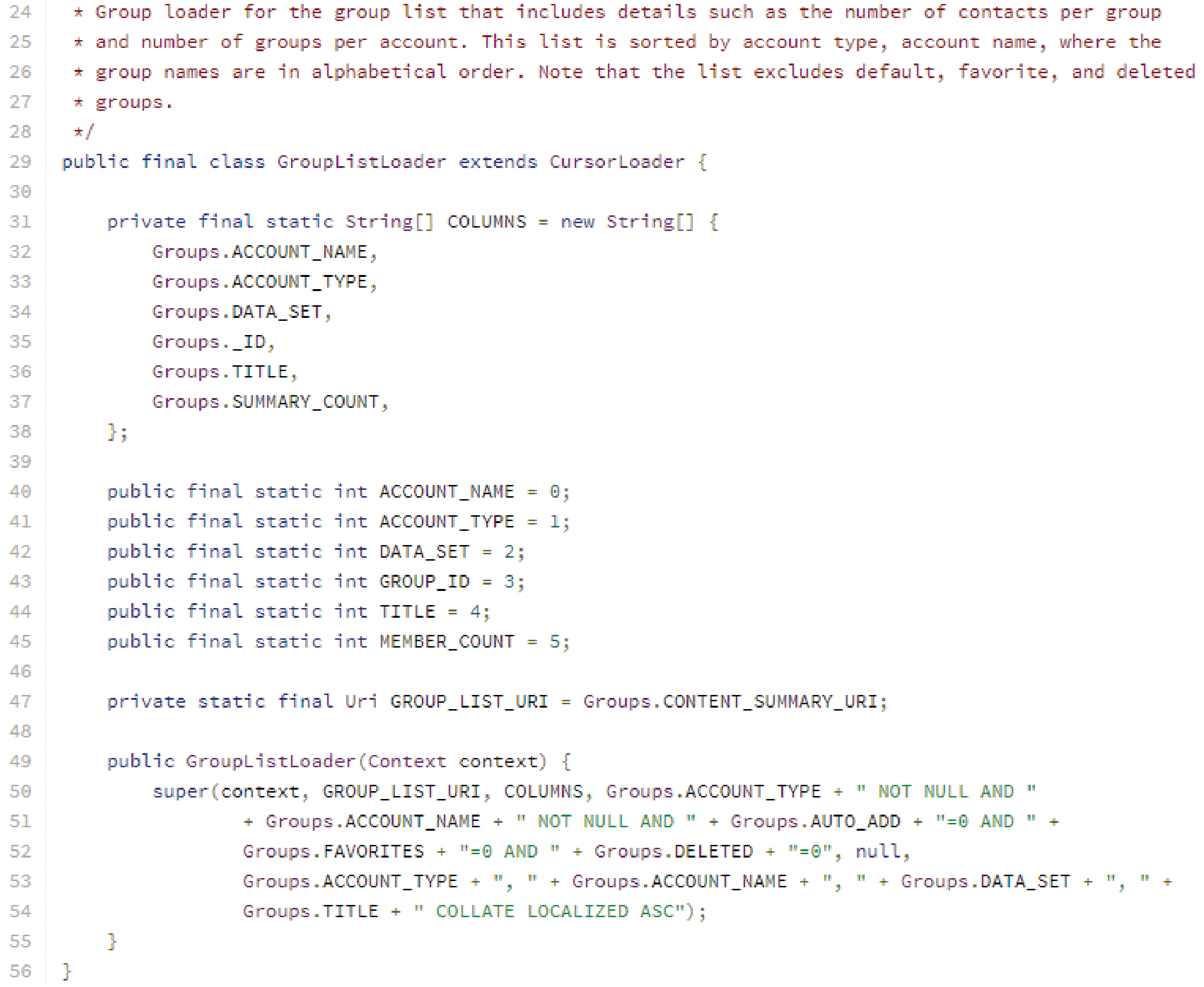


```
public final class GroupMemberLoader extends CursorLoader {

    public static class GroupEditorQuery {
        private static final String[] PROJECTION = new String[] {
            Data.CONTACT_ID,                      // 0
            Data.RAW_CONTACT_ID,                  // 1
            Data.DISPLAY_NAME_PRIMARY,            // 2
            Data.PHOTO_URI,                       // 3
            Data.LOOKUP_KEY,                      // 4
        };

        public static final int CONTACT_ID                   = 0;
        public static final int RAW_CONTACT_ID               = 1;
        public static final int CONTACT_DISPLAY_NAME_PRIMARY = 2;
        public static final int CONTACT_PHOTO_URI            = 3;
        public static final int CONTACT_LOOKUP_KEY           = 4;
    }

    public static class GroupDetailQuery {
        private static final String[] PROJECTION = new String[] {
            Data.CONTACT_ID,                      // 0
            Data.PHOTO_URI,                       // 1
            Data.LOOKUP_KEY,                      // 2
            Data.DISPLAY_NAME_PRIMARY,            // 3
            Data.CONTACT_PRESENCE,                // 4
            Data.CONTACT_STATUS,                  // 5
        };

        public static final int CONTACT_ID                   = 0;
        public static final int CONTACT_PHOTO_URI            = 1;
        public static final int CONTACT_LOOKUP_KEY           = 2;
        public static final int CONTACT_DISPLAY_NAME_PRIMARY = 3;
        public static final int CONTACT_PRESENCE_STATUS      = 4;
        public static final int CONTACT_STATUS               = 5;
    }

    private final long mGroupId;
```

https://android.googlesource.com/platform/packages/apps/Contacts/+/nougat-mr1-release/src/com/android/contacts/GroupMemberLoader.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>

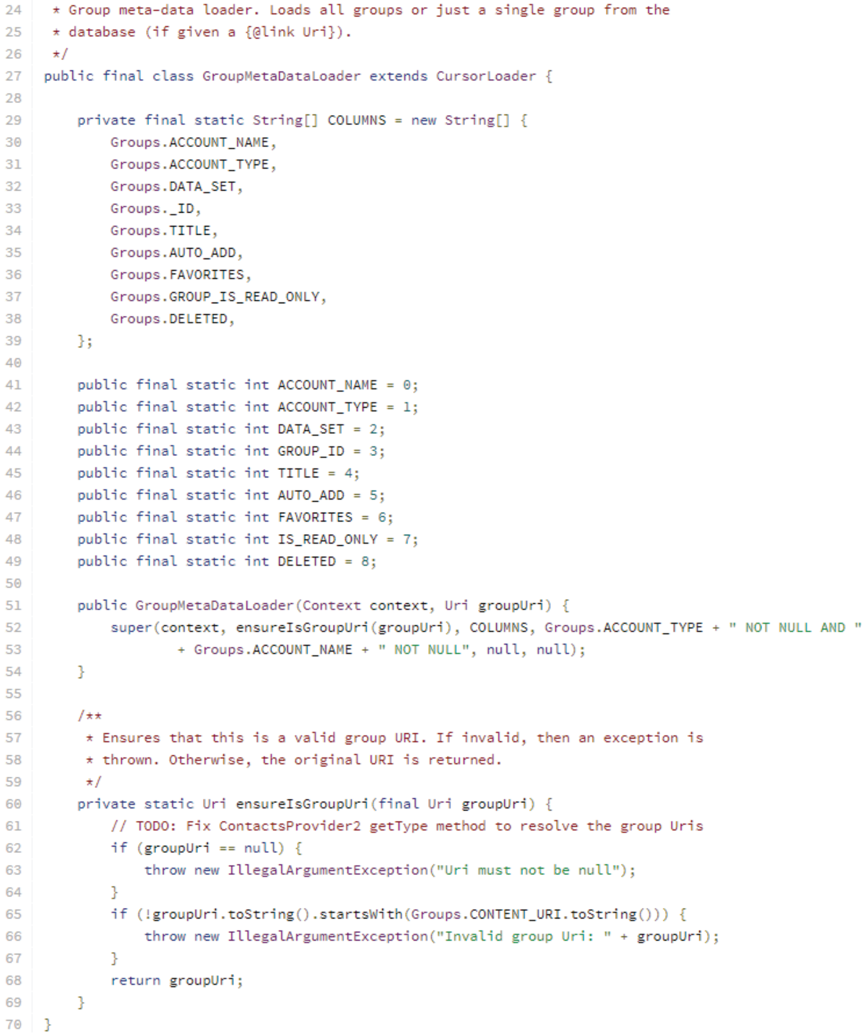


```
 * Group loader for the group list that includes details such as the number of contacts per group
 * and number of groups per account. This list is sorted by account type, account name, where the
 * group names are in alphabetical order. Note that the list excludes default, favorite, and deleted
 * groups.
 */
public final class GroupListLoader extends CursorLoader {

    private final static String[] COLUMNS = new String[] {
        Groups.ACCOUNT_NAME,
        Groups.ACCOUNT_TYPE,
        Groups.DATA_SET,
        Groups._ID,
        Groups.TITLE,
        Groups.SUMMARY_COUNT,
    };

    public final static int ACCOUNT_NAME = 0;
    public final static int ACCOUNT_TYPE = 1;
    public final static int DATA_SET = 2;
    public final static int GROUP_ID = 3;
    public final static int TITLE = 4;
    public final static int MEMBER_COUNT = 5;

    private static final Uri GROUP_LIST_URI = Groups.CONTENT_SUMMARY_URI;

    public GroupListLoader(Context context) {
        super(context, GROUP_LIST_URI, COLUMNS, Groups.ACCOUNT_TYPE + " NOT NULL AND "
                + Groups.ACCOUNT_NAME + " NOT NULL AND " + Groups.AUTO_ADD + "=0 AND " +
                Groups.FAVORITES + "=0 AND " + Groups.DELETED + "=0", null,
                Groups.ACCOUNT_TYPE + ", " + Groups.ACCOUNT_NAME + ", " + Groups.DATA_SET + ", " +
                Groups.TITLE + " COLLATE LOCALIZED ASC");
    }
}
```

https://android.googlesource.com/platform/packages/apps/Contacts/+/nougat-mr1-release/src/com/android/contacts/GroupListLoader.java

</td></tr>
</table>

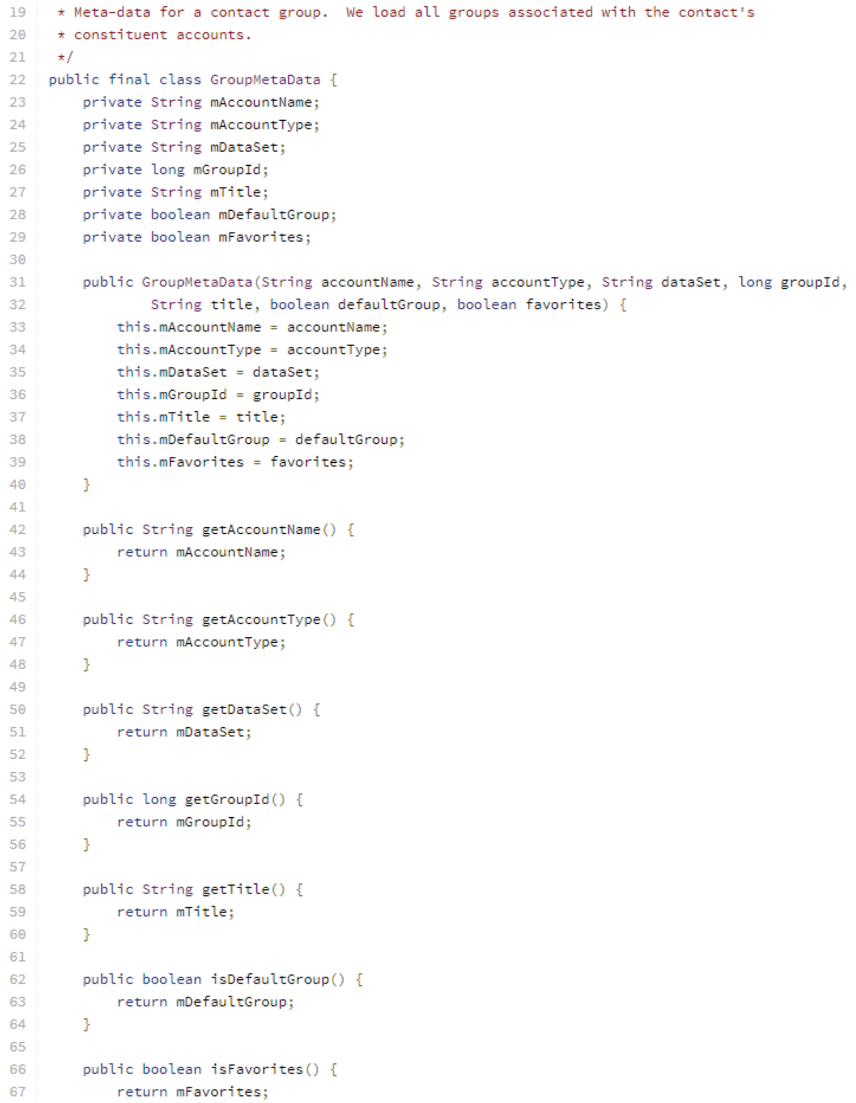
US9467838	Exemplary Supporting Evidence Regarding Accused Products



```
 * Group meta-data loader. Loads all groups or just a single group from the
 * database (if given a {@link Uri}).
 */
public final class GroupMetaDataLoader extends CursorLoader {

    private final static String[] COLUMNS = new String[] {
        Groups.ACCOUNT_NAME,
        Groups.ACCOUNT_TYPE,
        Groups.DATA_SET,
        Groups._ID,
        Groups.TITLE,
        Groups.AUTO_ADD,
        Groups.FAVORITES,
        Groups.GROUP_IS_READ_ONLY,
        Groups.DELETED,
    };

    public final static int ACCOUNT_NAME = 0;
    public final static int ACCOUNT_TYPE = 1;
    public final static int DATA_SET = 2;
    public final static int GROUP_ID = 3;
    public final static int TITLE = 4;
    public final static int AUTO_ADD = 5;
    public final static int FAVORITES = 6;
    public final static int IS_READ_ONLY = 7;
    public final static int DELETED = 8;

    public GroupMetaDataLoader(Context context, Uri groupUri) {
        super(context, ensureIsGroupUri(groupUri), COLUMNS, Groups.ACCOUNT_TYPE + " NOT NULL AND "
                + Groups.ACCOUNT_NAME + " NOT NULL", null, null);
    }

    /**
     * Ensures that this is a valid group URI. If invalid, then an exception is
     * thrown. Otherwise, the original URI is returned.
     */
    private static Uri ensureIsGroupUri(final Uri groupUri) {
        // TODO: Fix ContactsProvider2 getType method to resolve the group Uris
        if (groupUri == null) {
            throw new IllegalArgumentException("Uri must not be null");
        }
        if (!groupUri.toString().startsWith(Groups.CONTENT_URI.toString())) {
            throw new IllegalArgumentException("Invalid group Uri: " + groupUri);
        }
        return groupUri;
    }
}
```

https://android.googlesource.com/platform/packages/apps/Contacts/+/nougat-mr1-release/src/com/android/contacts/GroupMetaDataLoader.java

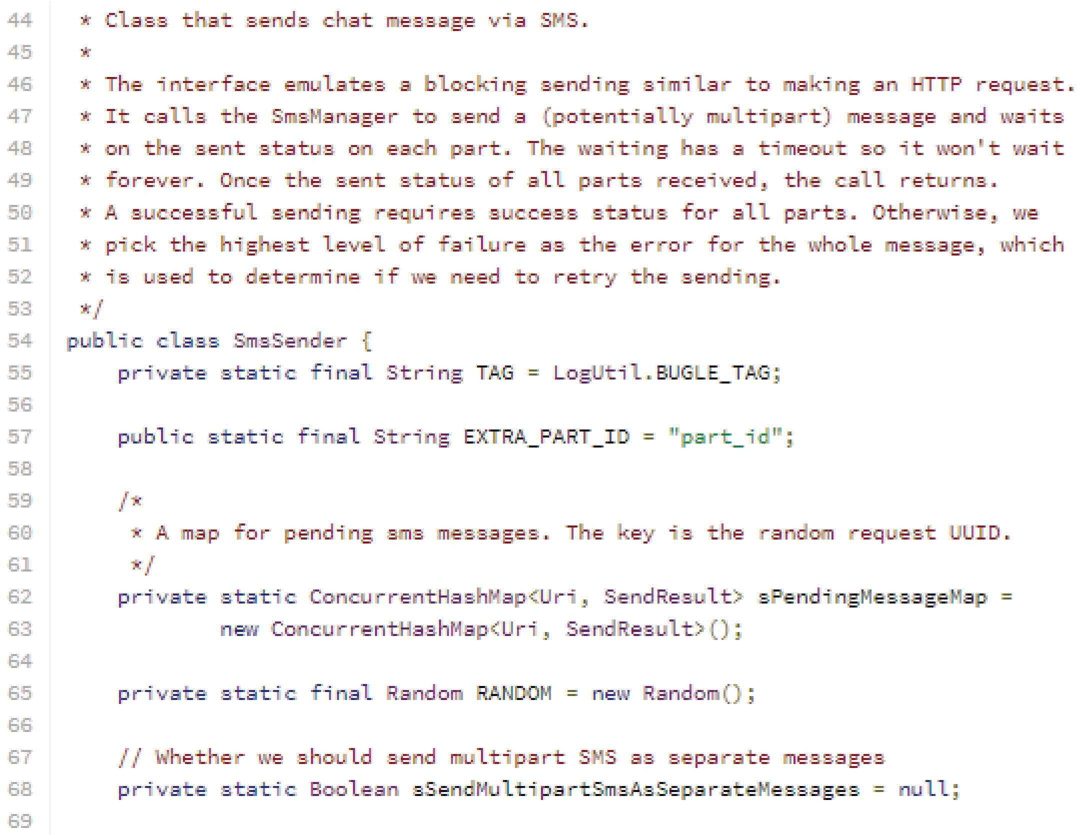
Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
 * Meta-data for a contact group.  We load all groups associated with the contact's
 * constituent accounts.
 */
public final class GroupMetaData {
    private String mAccountName;
    private String mAccountType;
    private String mDataSet;
    private long mGroupId;
    private String mTitle;
    private boolean mDefaultGroup;
    private boolean mFavorites;

    public GroupMetaData(String accountName, String accountType, String dataSet, long groupId,
            String title, boolean defaultGroup, boolean favorites) {
        this.mAccountName = accountName;
        this.mAccountType = accountType;
        this.mDataSet = dataSet;
        this.mGroupId = groupId;
        this.mTitle = title;
        this.mDefaultGroup = defaultGroup;
        this.mFavorites = favorites;
    }

    public String getAccountName() {
        return mAccountName;
    }

    public String getAccountType() {
        return mAccountType;
    }

    public String getDataSet() {
        return mDataSet;
    }

    public long getGroupId() {
        return mGroupId;
    }

    public String getTitle() {
        return mTitle;
    }

    public boolean isDefaultGroup() {
        return mDefaultGroup;
    }

    public boolean isFavorites() {
        return mFavorites;
```

https://android.googlesource.com/platform/packages/apps/ContactsCommon/+/nougat-mr1-release/src/com/android/contacts/common/GroupMetaData.java

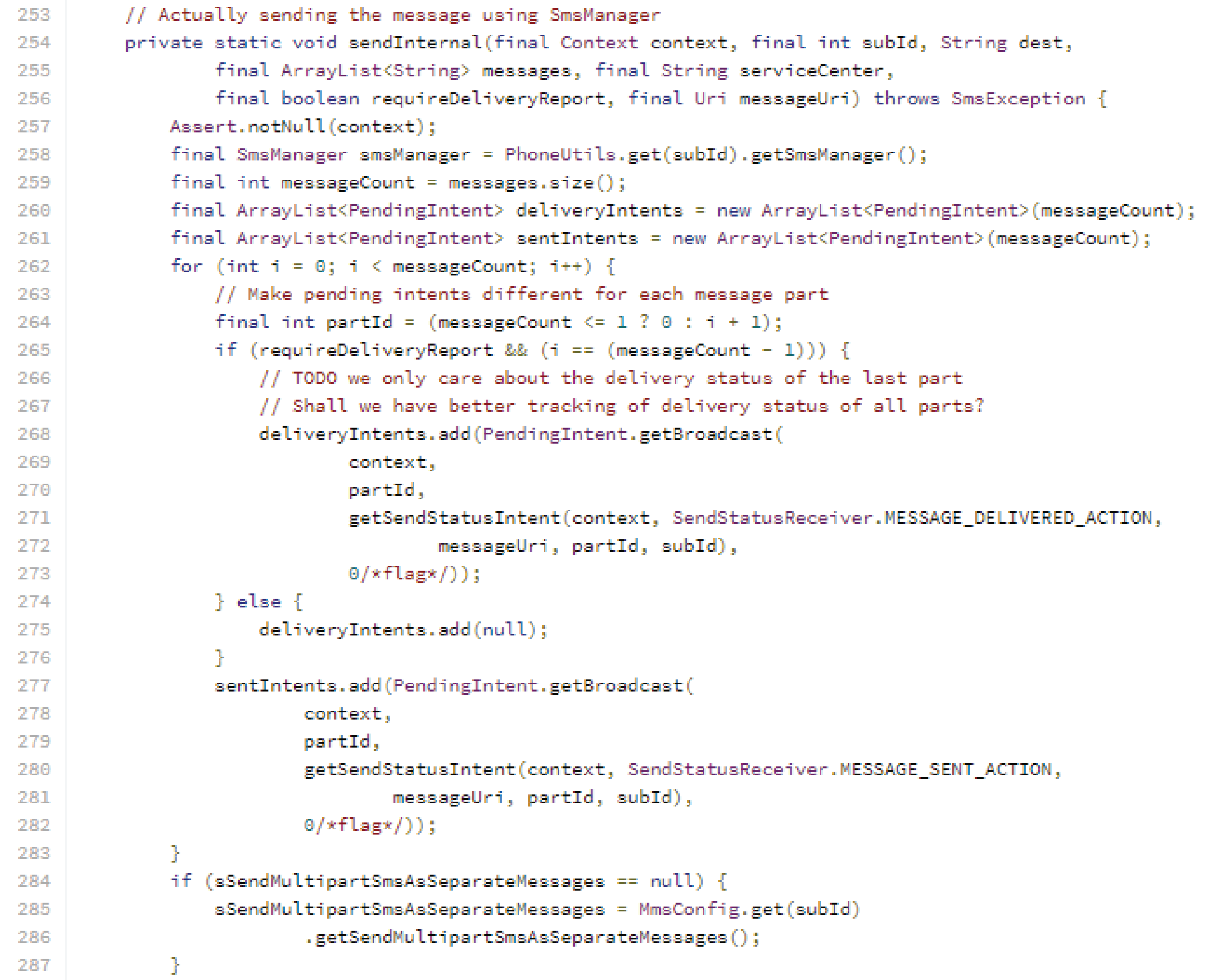
Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
 * Class that sends chat message via SMS.
 *
 * The interface emulates a blocking sending similar to making an HTTP request.
 * It calls the SmsManager to send a (potentially multipart) message and waits
 * on the sent status on each part. The waiting has a timeout so it won't wait
 * forever. Once the sent status of all parts received, the call returns.
 * A successful sending requires success status for all parts. Otherwise, we
 * pick the highest level of failure as the error for the whole message, which
 * is used to determine if we need to retry the sending.
 */
public class SmsSender {
    private static final String TAG = LogUtil.BUGLE_TAG;

    public static final String EXTRA_PART_ID = "part_id";

    /*
     * A map for pending sms messages. The key is the random request UUID.
     */
    private static ConcurrentHashMap<Uri, SendResult> sPendingMessageMap =
            new ConcurrentHashMap<Uri, SendResult>();

    private static final Random RANDOM = new Random();

    // Whether we should send multipart SMS as separate messages
    private static Boolean sSendMultipartSmsAsSeparateMessages = null;
```

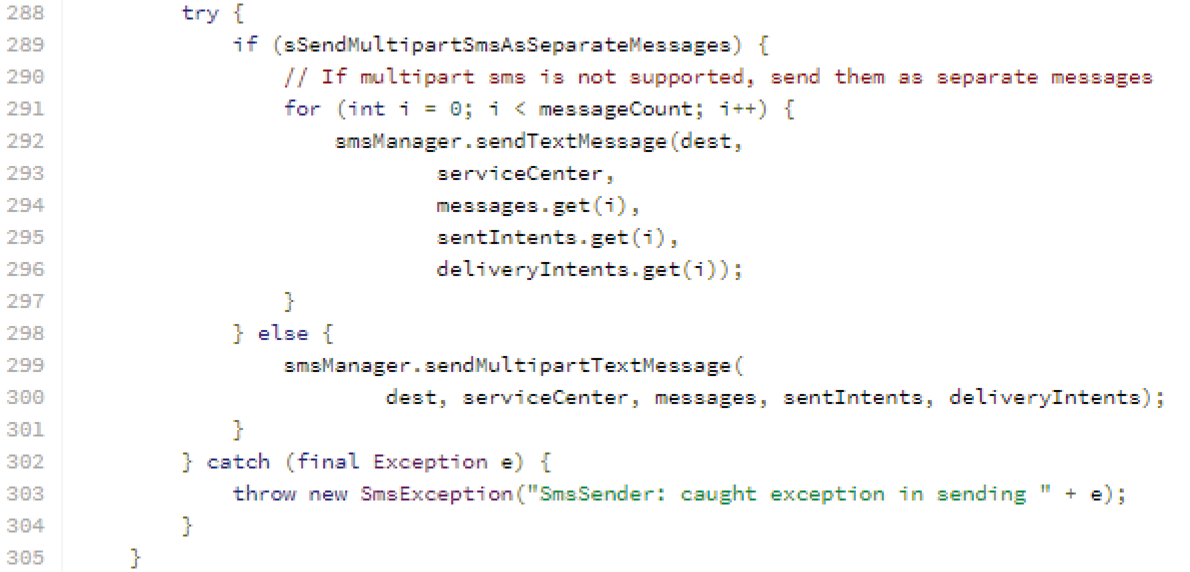
https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/SmsSender.java

US9467838	Exemplary Supporting Evidence Regarding Accused Products



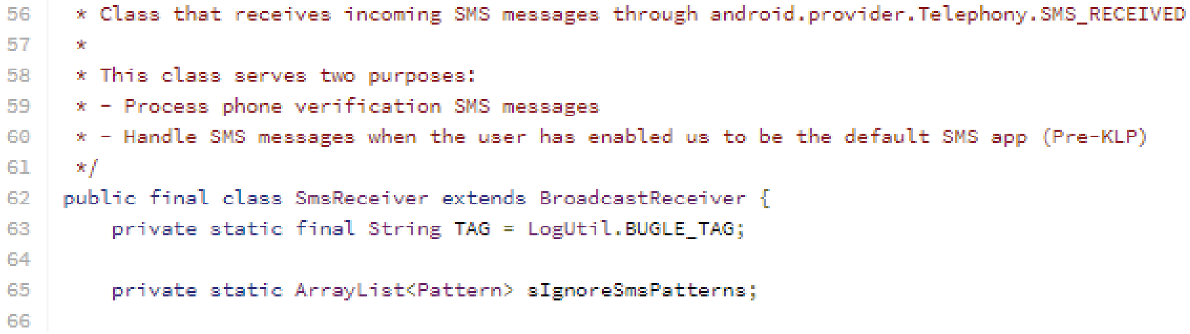
```
    // Actually sending the message using SmsManager
    private static void sendInternal(final Context context, final int subId, String dest,
            final ArrayList<String> messages, final String serviceCenter,
            final boolean requireDeliveryReport, final Uri messageUri) throws SmsException {
        Assert.notNull(context);
        final SmsManager smsManager = PhoneUtils.get(subId).getSmsManager();
        final int messageCount = messages.size();
        final ArrayList<PendingIntent> deliveryIntents = new ArrayList<PendingIntent>(messageCount);
        final ArrayList<PendingIntent> sentIntents = new ArrayList<PendingIntent>(messageCount);
        for (int i = 0; i < messageCount; i++) {
            // Make pending intents different for each message part
            final int partId = (messageCount <= 1 ? 0 : i + 1);
            if (requireDeliveryReport && (i == (messageCount - 1))) {
                // TODO we only care about the delivery status of the last part
                // Shall we have better tracking of delivery status of all parts?
                deliveryIntents.add(PendingIntent.getBroadcast(
                        context,
                        partId,
                        getSendStatusIntent(context, SendStatusReceiver.MESSAGE_DELIVERED_ACTION,
                                messageUri, partId, subId),
                        0/*flag*/));
            } else {
                deliveryIntents.add(null);
            }
            sentIntents.add(PendingIntent.getBroadcast(
                    context,
                    partId,
                    getSendStatusIntent(context, SendStatusReceiver.MESSAGE_SENT_ACTION,
                            messageUri, partId, subId),
                    0/*flag*/));
        }
        if (sSendMultipartSmsAsSeparateMessages == null) {
            sSendMultipartSmsAsSeparateMessages = MmsConfig.get(subId)
                    .getSendMultipartSmsAsSeparateMessages();
        }
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/SmsSender.java

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
        try {
            if (sSendMultipartSmsAsSeparateMessages) {
                // If multipart sms is not supported, send them as separate messages
                for (int i = 0; i < messageCount; i++) {
                    smsManager.sendTextMessage(dest,
                            serviceCenter,
                            messages.get(i),
                            sentIntents.get(i),
                            deliveryIntents.get(i));
                }
            } else {
                smsManager.sendMultipartTextMessage(
                        dest, serviceCenter, messages, sentIntents, deliveryIntents);
            }
        } catch (final Exception e) {
            throw new SmsException("SmsSender: caught exception in sending " + e);
        }
    }
```

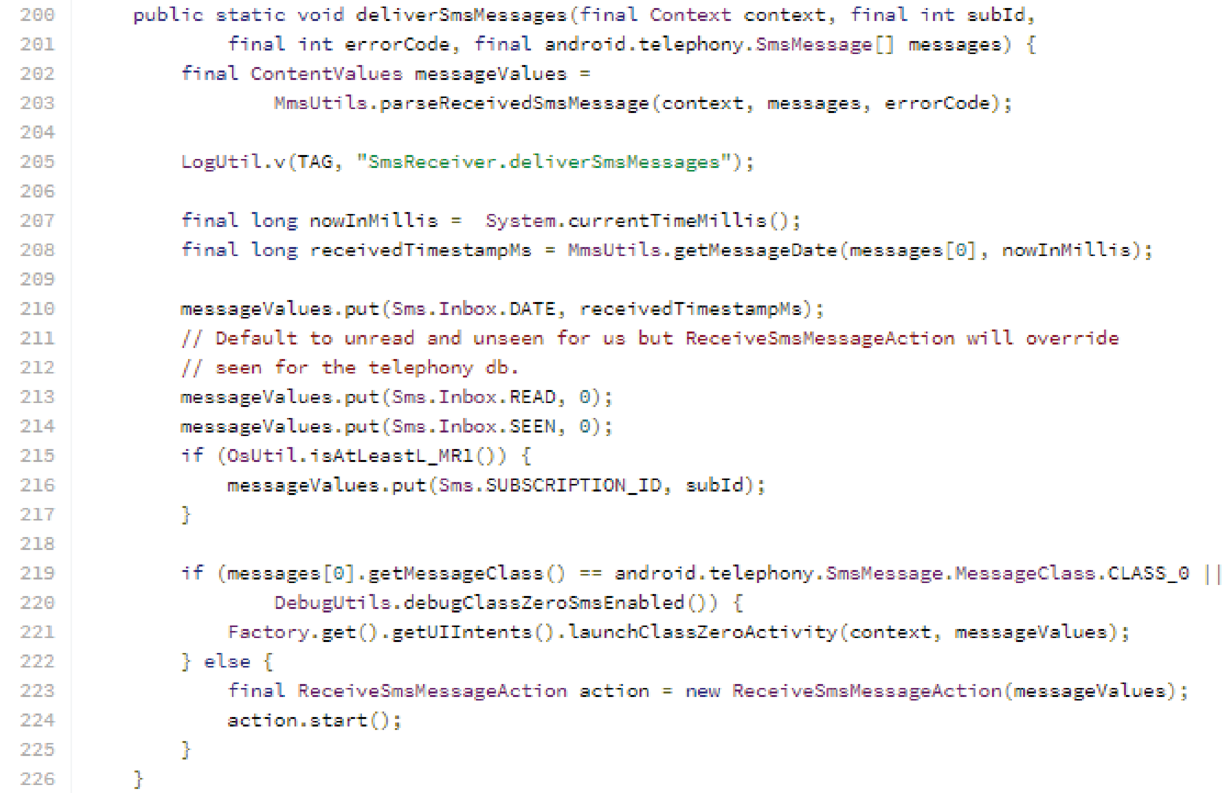
https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/SmsSender.java



```
 * Class that receives incoming SMS messages through android.provider.Telephony.SMS_RECEIVED
 *
 * This class serves two purposes:
 * - Process phone verification SMS messages
 * - Handle SMS messages when the user has enabled us to be the default SMS app (Pre-KLP)
 */
public final class SmsReceiver extends BroadcastReceiver {
    private static final String TAG = LogUtil.BUGLE_TAG;

    private static ArrayList<Pattern> sIgnoreSmsPatterns;

```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/receiver/SmsReceiver.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

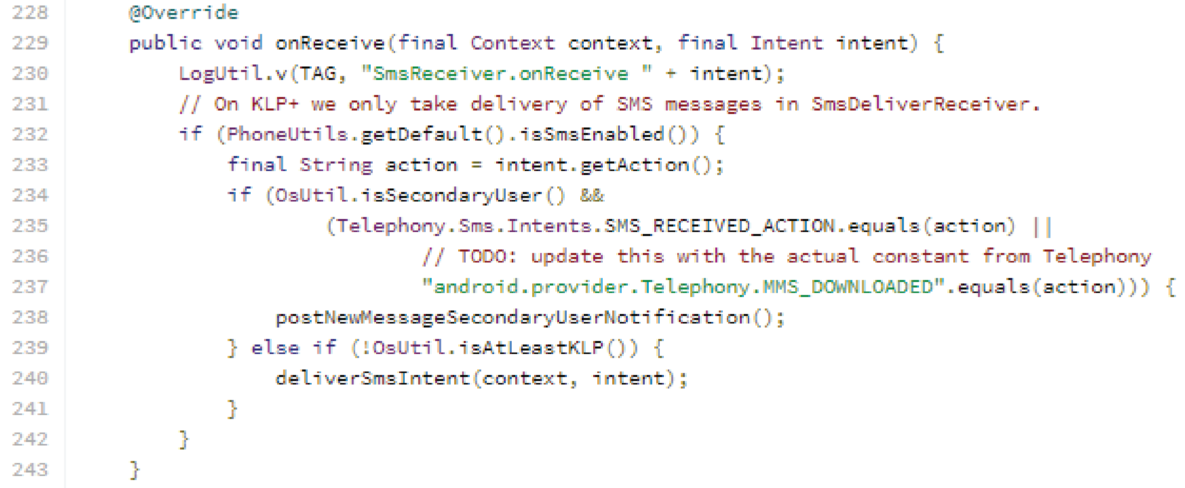
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
public static void deliverSmsMessages(final Context context, final int subId,
        final int errorCode, final android.telephony.SmsMessage[] messages) {
    final ContentValues messageValues =
            MmsUtils.parseReceivedSmsMessage(context, messages, errorCode);

    LogUtil.v(TAG, "SmsReceiver.deliverSmsMessages");

    final long nowInMillis =  System.currentTimeMillis();
    final long receivedTimestampMs = MmsUtils.getMessageDate(messages[0], nowInMillis);

    messageValues.put(Sms.Inbox.DATE, receivedTimestampMs);
    // Default to unread and unseen for us but ReceiveSmsMessageAction will override
    // seen for the telephony db.
    messageValues.put(Sms.Inbox.READ, 0);
    messageValues.put(Sms.Inbox.SEEN, 0);
    if (OsUtil.isAtLeastL_MR1()) {
        messageValues.put(Sms.SUBSCRIPTION_ID, subId);
    }

    if (messages[0].getMessageClass() == android.telephony.SmsMessage.MessageClass.CLASS_0 ||
            DebugUtils.debugClassZeroSmsEnabled()) {
        Factory.get().getUIIntents().launchClassZeroActivity(context, messageValues);
    } else {
        final ReceiveSmsMessageAction action = new ReceiveSmsMessageAction(messageValues);
        action.start();
    }
}
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/receiver/SmsReceiver.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

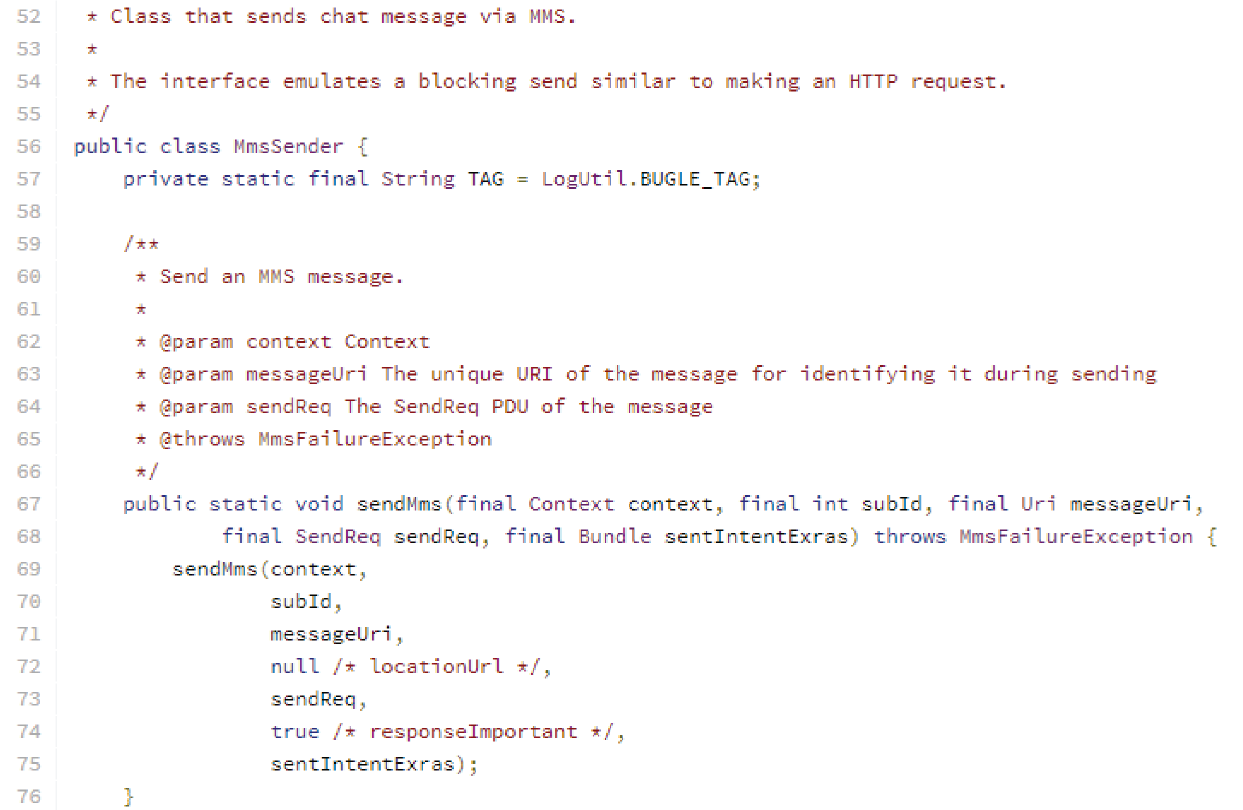
US9467838	Exemplary Supporting Evidence Regarding Accused Products



```
@Override
public void onReceive(final Context context, final Intent intent) {
    LogUtil.v(TAG, "SmsReceiver.onReceive " + intent);
    // On KLP+ we only take delivery of SMS messages in SmsDeliverReceiver.
    if (PhoneUtils.getDefault().isSmsEnabled()) {
        final String action = intent.getAction();
        if (OsUtil.isSecondaryUser() &&
                (Telephony.Sms.Intents.SMS_RECEIVED_ACTION.equals(action) ||
                        // TODO: update this with the actual constant from Telephony
                        "android.provider.Telephony.MMS_DOWNLOADED".equals(action))) {
            postNewMessageSecondaryUserNotification();
        } else if (!OsUtil.isAtLeastKLP()) {
            deliverSmsIntent(context, intent);
        }
    }
}
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/receiver/SmsReceiver.java

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>

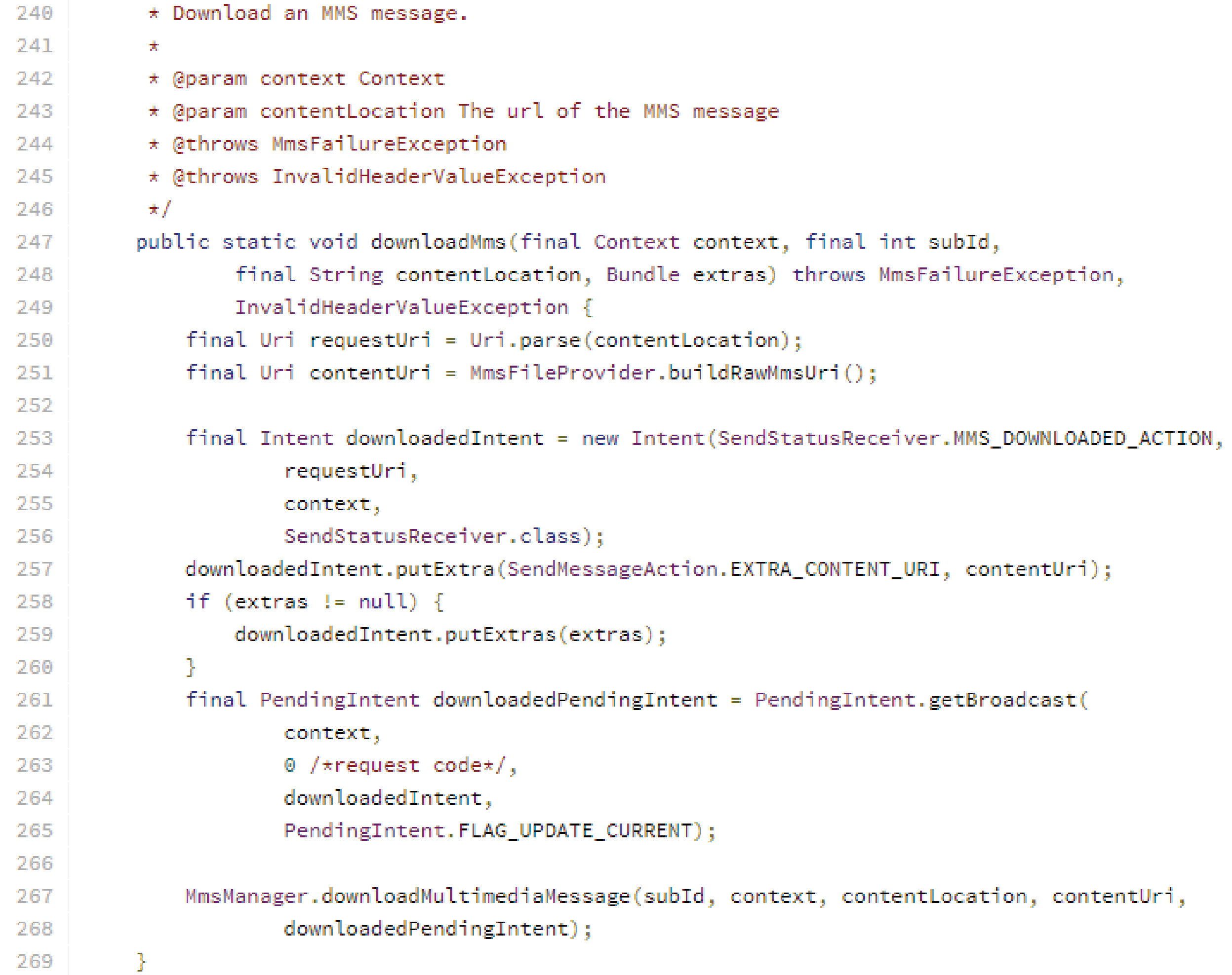


```
 * Class that sends chat message via MMS.
 *
 * The interface emulates a blocking send similar to making an HTTP request.
 */
public class MmsSender {
    private static final String TAG = LogUtil.BUGLE_TAG;

    /**
     * Send an MMS message.
     *
     * @param context Context
     * @param messageUri The unique URI of the message for identifying it during sending
     * @param sendReq The SendReq PDU of the message
     * @throws MmsFailureException
     */
    public static void sendMms(final Context context, final int subId, final Uri messageUri,
            final SendReq sendReq, final Bundle sentIntentExras) throws MmsFailureException {
        sendMms(context,
                subId,
                messageUri,
                null /* locationUrl */,
                sendReq,
                true /* responseImportant */,
                sentIntentExras);
    }
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/MmsSender.java

</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

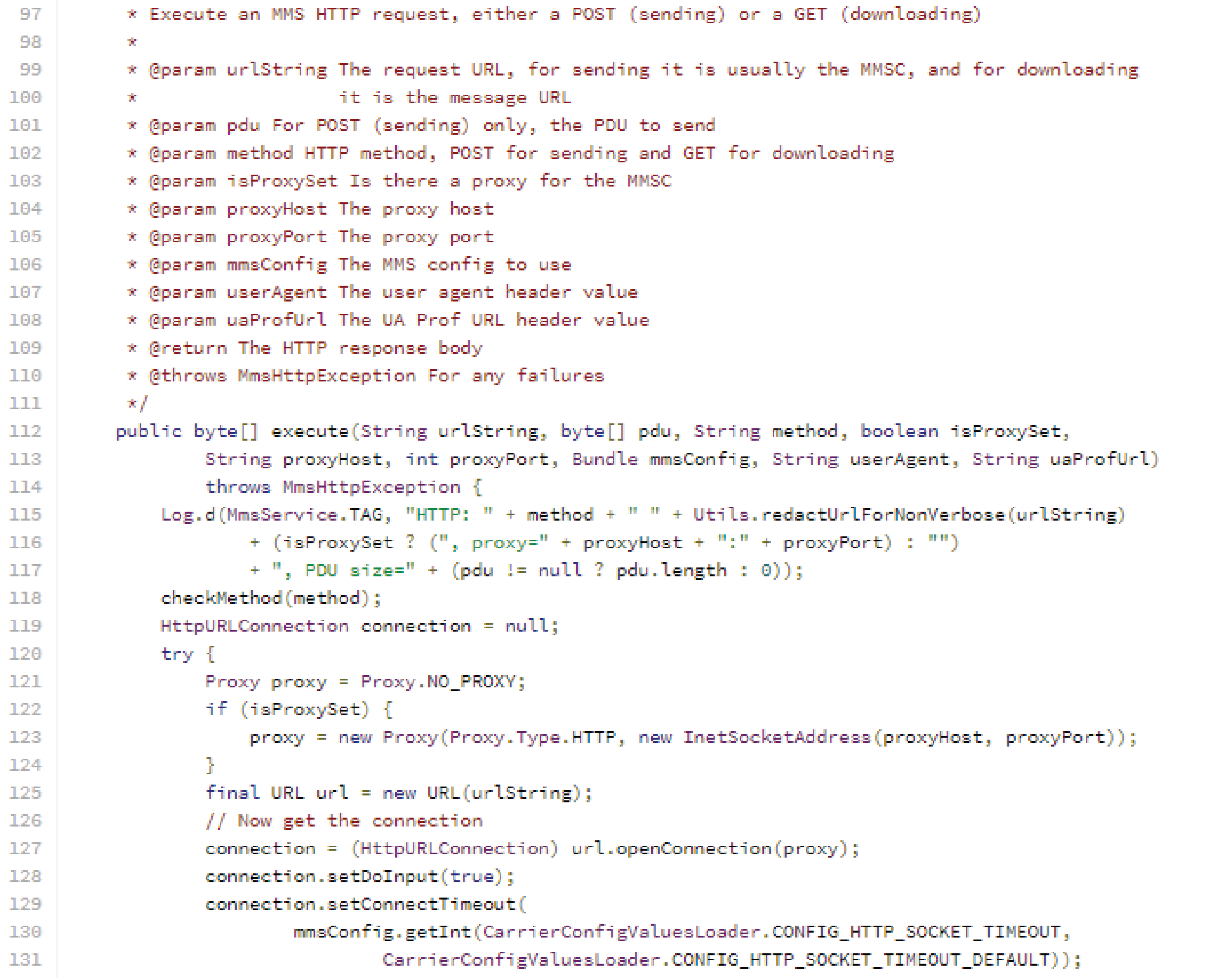
<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>
<pre>
     * Download an MMS message.
     *
     * @param context Context
     * @param contentLocation The url of the MMS message
     * @throws MmsFailureException
     * @throws InvalidHeaderValueException
     */
    public static void downloadMms(final Context context, final int subId,
            final String contentLocation, Bundle extras) throws MmsFailureException,
            InvalidHeaderValueException {
        final Uri requestUri = Uri.parse(contentLocation);
        final Uri contentUri = MmsFileProvider.buildRawMmsUri();

        final Intent downloadedIntent = new Intent(SendStatusReceiver.MMS_DOWNLOADED_ACTION,
                requestUri,
                context,
                SendStatusReceiver.class);
        downloadedIntent.putExtra(SendMessageAction.EXTRA_CONTENT_URI, contentUri);
        if (extras != null) {
            downloadedIntent.putExtras(extras);
        }
        final PendingIntent downloadedPendingIntent = PendingIntent.getBroadcast(
                context,
                0 /*request code*/,
                downloadedIntent,
                PendingIntent.FLAG_UPDATE_CURRENT);

        MmsManager.downloadMultimediaMessage(subId, context, contentLocation, contentUri,
                downloadedPendingIntent);
    }
</pre>
https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/MmsSender.java</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

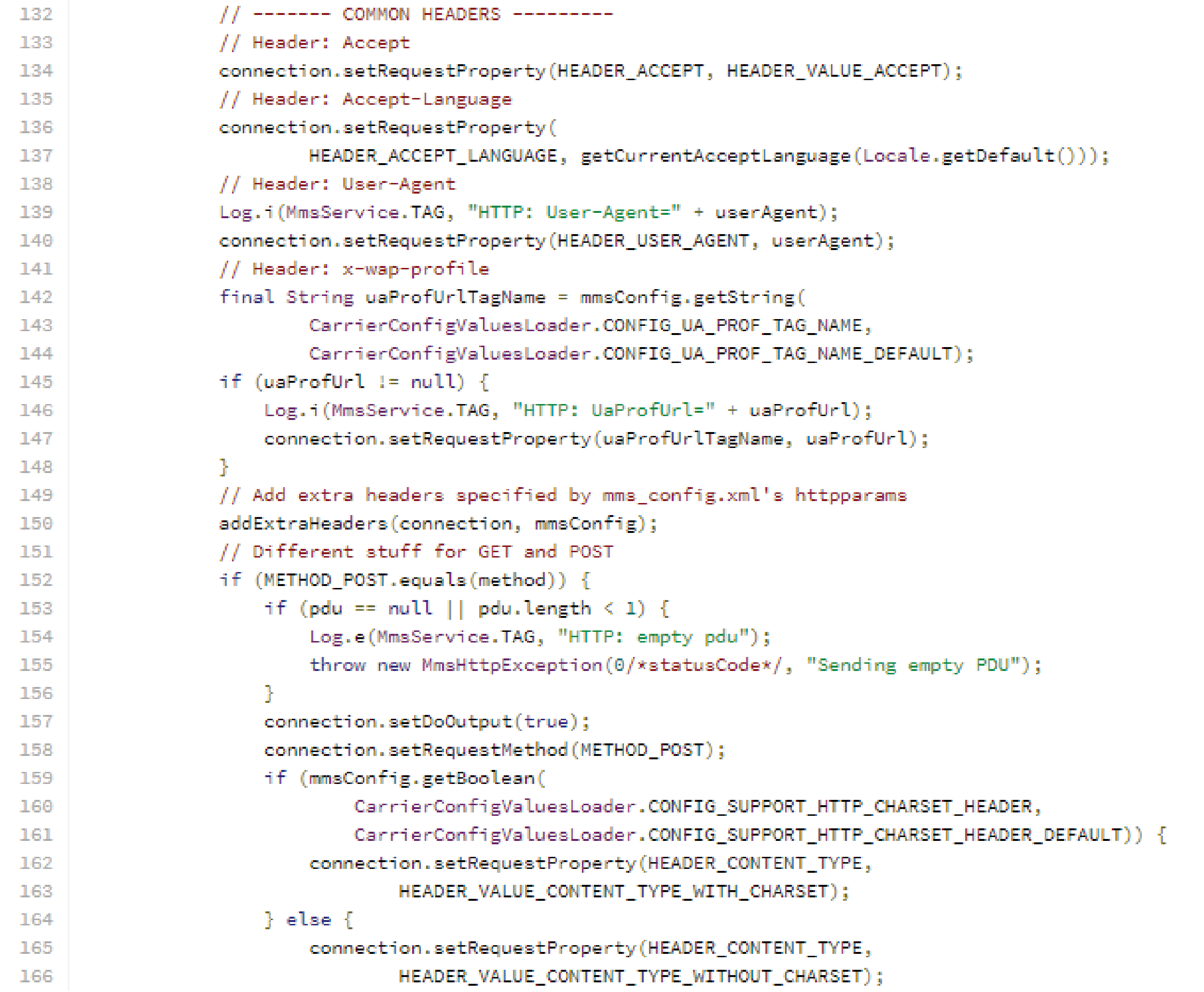
US9467838	Exemplary Supporting Evidence Regarding Accused Products



```
     * Execute an MMS HTTP request, either a POST (sending) or a GET (downloading)
     *
     * @param urlString The request URL, for sending it is usually the MMSC, and for downloading
     *                  it is the message URL
     * @param pdu For POST (sending) only, the PDU to send
     * @param method HTTP method, POST for sending and GET for downloading
     * @param isProxySet Is there a proxy for the MMSC
     * @param proxyHost The proxy host
     * @param proxyPort The proxy port
     * @param mmsConfig The MMS config to use
     * @param userAgent The user agent header value
     * @param uaProfUrl The UA Prof URL header value
     * @return The HTTP response body
     * @throws MmsHttpException For any failures
     */
    public byte[] execute(String urlString, byte[] pdu, String method, boolean isProxySet,
            String proxyHost, int proxyPort, Bundle mmsConfig, String userAgent, String uaProfUrl)
            throws MmsHttpException {
        Log.d(MmsService.TAG, "HTTP: " + method + " " + Utils.redactUrlForNonVerbose(urlString)
                + (isProxySet ? (", proxy=" + proxyHost + ":" + proxyPort) : "")
                + ", PDU size=" + (pdu != null ? pdu.length : 0));
        checkMethod(method);
        HttpURLConnection connection = null;
        try {
            Proxy proxy = Proxy.NO_PROXY;
            if (isProxySet) {
                proxy = new Proxy(Proxy.Type.HTTP, new InetSocketAddress(proxyHost, proxyPort));
            }
            final URL url = new URL(urlString);
            // Now get the connection
            connection = (HttpURLConnection) url.openConnection(proxy);
            connection.setDoInput(true);
            connection.setConnectTimeout(
                    mmsConfig.getInt(CarrierConfigValuesLoader.CONFIG_HTTP_SOCKET_TIMEOUT,
                            CarrierConfigValuesLoader.CONFIG_HTTP_SOCKET_TIMEOUT_DEFAULT));
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/android/support/v7/mms/MmsHttpClient.java

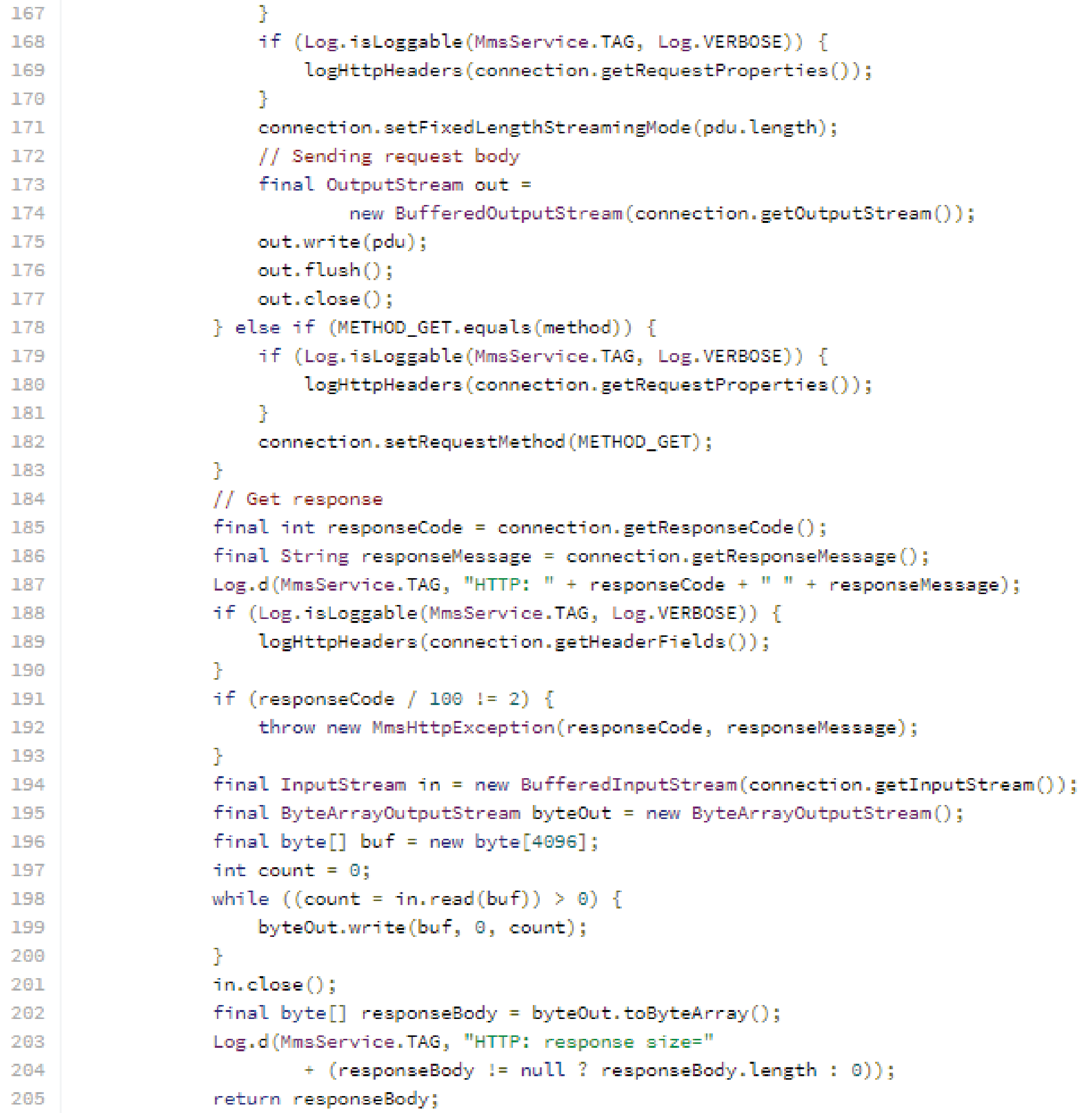
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	(code screenshot and source below)



```
        // ------- COMMON HEADERS ---------
        // Header: Accept
        connection.setRequestProperty(HEADER_ACCEPT, HEADER_VALUE_ACCEPT);
        // Header: Accept-Language
        connection.setRequestProperty(
                HEADER_ACCEPT_LANGUAGE, getCurrentAcceptLanguage(Locale.getDefault()));
        // Header: User-Agent
        Log.i(MmsService.TAG, "HTTP: User-Agent=" + userAgent);
        connection.setRequestProperty(HEADER_USER_AGENT, userAgent);
        // Header: x-wap-profile
        final String uaProfUrlTagName = mmsConfig.getString(
                CarrierConfigValuesLoader.CONFIG_UA_PROF_TAG_NAME,
                CarrierConfigValuesLoader.CONFIG_UA_PROF_TAG_NAME_DEFAULT);
        if (uaProfUrl != null) {
            Log.i(MmsService.TAG, "HTTP: UaProfUrl=" + uaProfUrl);
            connection.setRequestProperty(uaProfUrlTagName, uaProfUrl);
        }
        // Add extra headers specified by mms_config.xml's httpparams
        addExtraHeaders(connection, mmsConfig);
        // Different stuff for GET and POST
        if (METHOD_POST.equals(method)) {
            if (pdu == null || pdu.length < 1) {
                Log.e(MmsService.TAG, "HTTP: empty pdu");
                throw new MmsHttpException(0/*statusCode*/, "Sending empty PDU");
            }
            connection.setDoOutput(true);
            connection.setRequestMethod(METHOD_POST);
            if (mmsConfig.getBoolean(
                    CarrierConfigValuesLoader.CONFIG_SUPPORT_HTTP_CHARSET_HEADER,
                    CarrierConfigValuesLoader.CONFIG_SUPPORT_HTTP_CHARSET_HEADER_DEFAULT)) {
                connection.setRequestProperty(HEADER_CONTENT_TYPE,
                        HEADER_VALUE_CONTENT_TYPE_WITH_CHARSET);
            } else {
                connection.setRequestProperty(HEADER_CONTENT_TYPE,
                        HEADER_VALUE_CONTENT_TYPE_WITHOUT_CHARSET);
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/android/support/v7/mms/MmsHttpClient.java

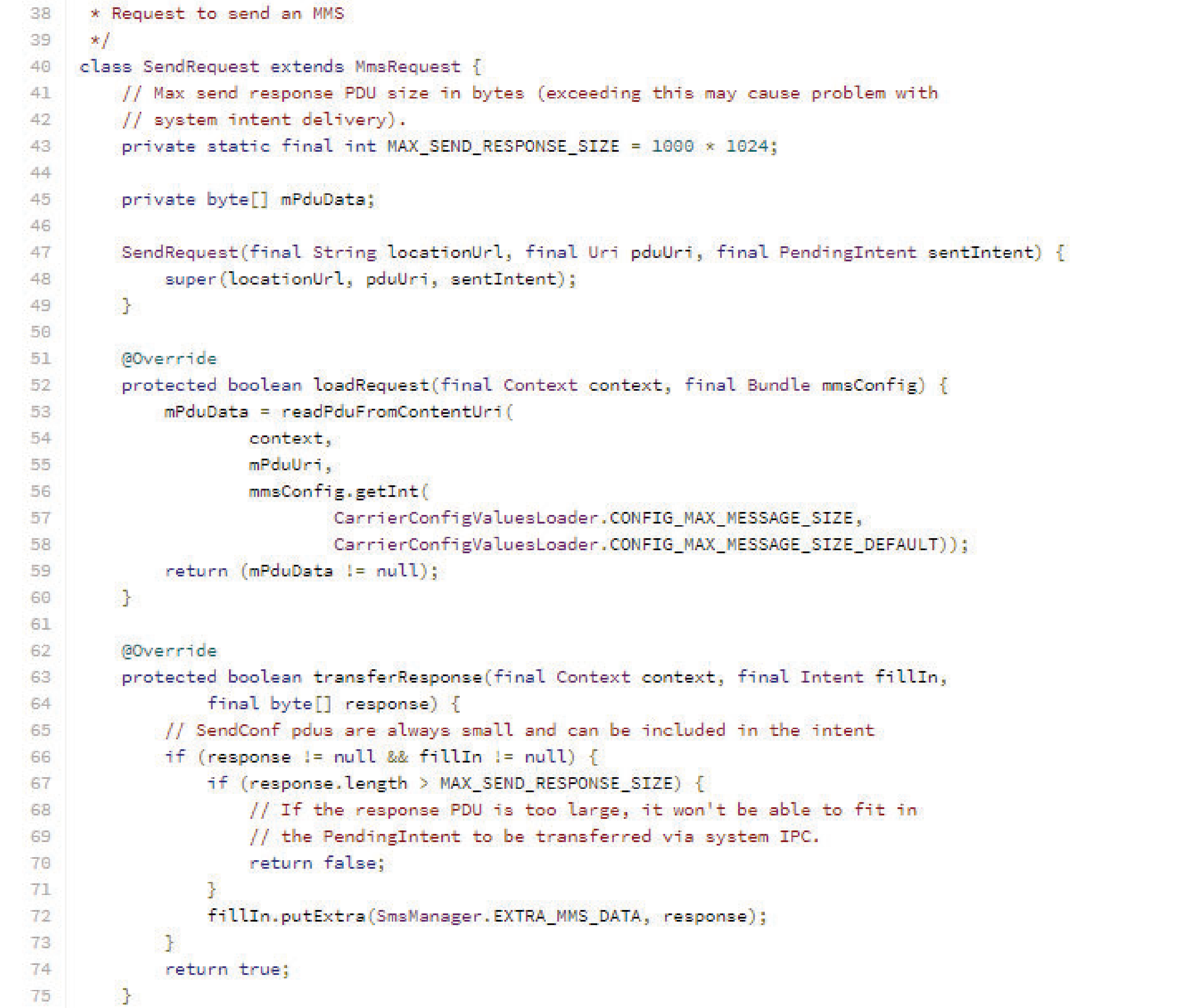
Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
            }
            if (Log.isLoggable(MmsService.TAG, Log.VERBOSE)) {
                logHttpHeaders(connection.getRequestProperties());
            }
            connection.setFixedLengthStreamingMode(pdu.length);
            // Sending request body
            final OutputStream out =
                    new BufferedOutputStream(connection.getOutputStream());
            out.write(pdu);
            out.flush();
            out.close();
        } else if (METHOD_GET.equals(method)) {
            if (Log.isLoggable(MmsService.TAG, Log.VERBOSE)) {
                logHttpHeaders(connection.getRequestProperties());
            }
            connection.setRequestMethod(METHOD_GET);
        }
        // Get response
        final int responseCode = connection.getResponseCode();
        final String responseMessage = connection.getResponseMessage();
        Log.d(MmsService.TAG, "HTTP: " + responseCode + " " + responseMessage);
        if (Log.isLoggable(MmsService.TAG, Log.VERBOSE)) {
            logHttpHeaders(connection.getHeaderFields());
        }
        if (responseCode / 100 != 2) {
            throw new MmsHttpException(responseCode, responseMessage);
        }
        final InputStream in = new BufferedInputStream(connection.getInputStream());
        final ByteArrayOutputStream byteOut = new ByteArrayOutputStream();
        final byte[] buf = new byte[4096];
        int count = 0;
        while ((count = in.read(buf)) > 0) {
            byteOut.write(buf, 0, count);
        }
        in.close();
        final byte[] responseBody = byteOut.toByteArray();
        Log.d(MmsService.TAG, "HTTP: response size="
                + (responseBody != null ? responseBody.length : 0));
        return responseBody;
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/android/support/v7/mms/MmsHttpClient.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
 * Request to send an MMS
 */
class SendRequest extends MmsRequest {
    // Max send response PDU size in bytes (exceeding this may cause problem with
    // system intent delivery).
    private static final int MAX_SEND_RESPONSE_SIZE = 1000 * 1024;

    private byte[] mPduData;

    SendRequest(final String locationUrl, final Uri pduUri, final PendingIntent sentIntent) {
        super(locationUrl, pduUri, sentIntent);
    }

    @Override
    protected boolean loadRequest(final Context context, final Bundle mmsConfig) {
        mPduData = readPduFromContentUri(
                context,
                mPduUri,
                mmsConfig.getInt(
                        CarrierConfigValuesLoader.CONFIG_MAX_MESSAGE_SIZE,
                        CarrierConfigValuesLoader.CONFIG_MAX_MESSAGE_SIZE_DEFAULT));
        return (mPduData != null);
    }

    @Override
    protected boolean transferResponse(final Context context, final Intent fillIn,
            final byte[] response) {
        // SendConf pdus are always small and can be included in the intent
        if (response != null && fillIn != null) {
            if (response.length > MAX_SEND_RESPONSE_SIZE) {
                // If the response PDU is too large, it won't be able to fit in
                // the PendingIntent to be transferred via system IPC.
                return false;
            }
            fillIn.putExtra(SmsManager.EXTRA_MMS_DATA, response);
        }
        return true;
    }
```

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

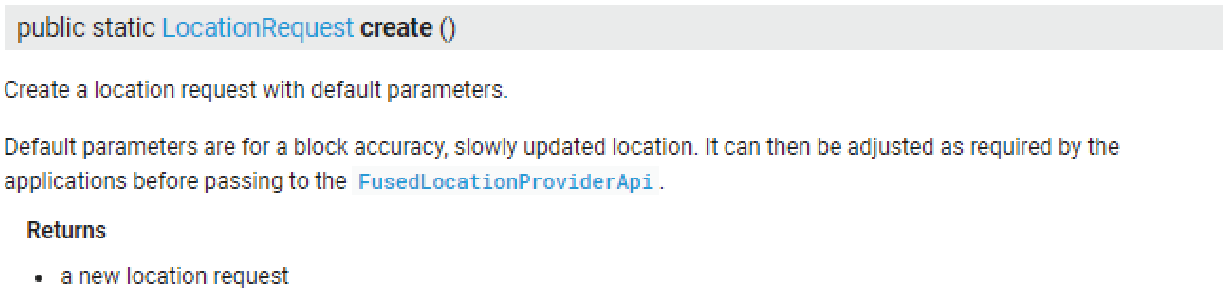
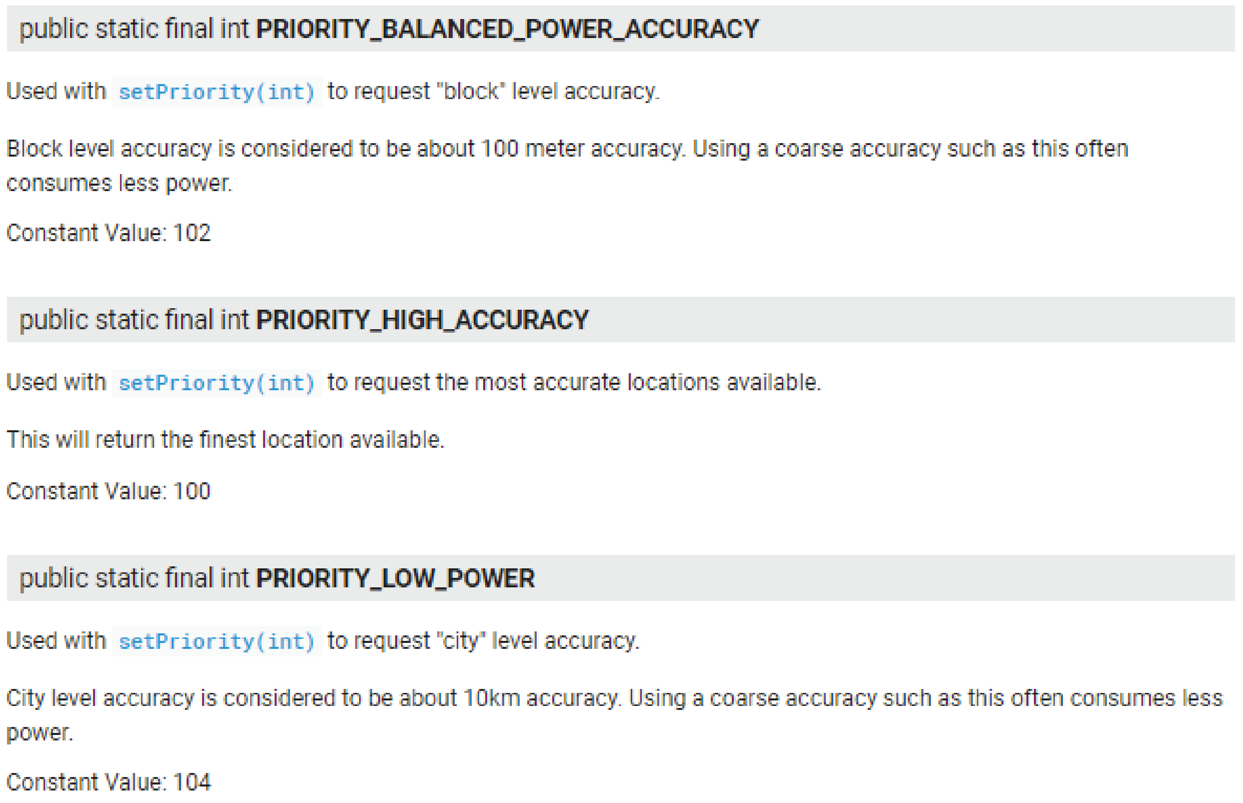
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/android/support/v7/mms/SendRequest.java  public static LocationRequest **create** () Create a location request with default parameters. Default parameters are for a block accuracy, slowly updated location. It can then be adjusted as required by the applications before passing to the `FusedLocationProviderApi` . **Returns** - a new location request https://developers.google.com/android/reference/com/google/android/gms/location/LocationRequest

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public static final int **PRIORITY_BALANCED_POWER_ACCURACY** Used with `setPriority(int)` to request "block" level accuracy. Block level accuracy is considered to be about 100 meter accuracy. Using a coarse accuracy such as this often consumes less power. Constant Value: 102 public static final int **PRIORITY_HIGH_ACCURACY** Used with `setPriority(int)` to request the most accurate locations available. This will return the finest location available. Constant Value: 100 public static final int **PRIORITY_LOW_POWER** Used with `setPriority(int)` to request "city" level accuracy. City level accuracy is considered to be about 10km accuracy. Using a coarse accuracy such as this often consumes less power. Constant Value: 104 https://developers.google.com/android/reference/com/google/android/gms/location/LocationRequest

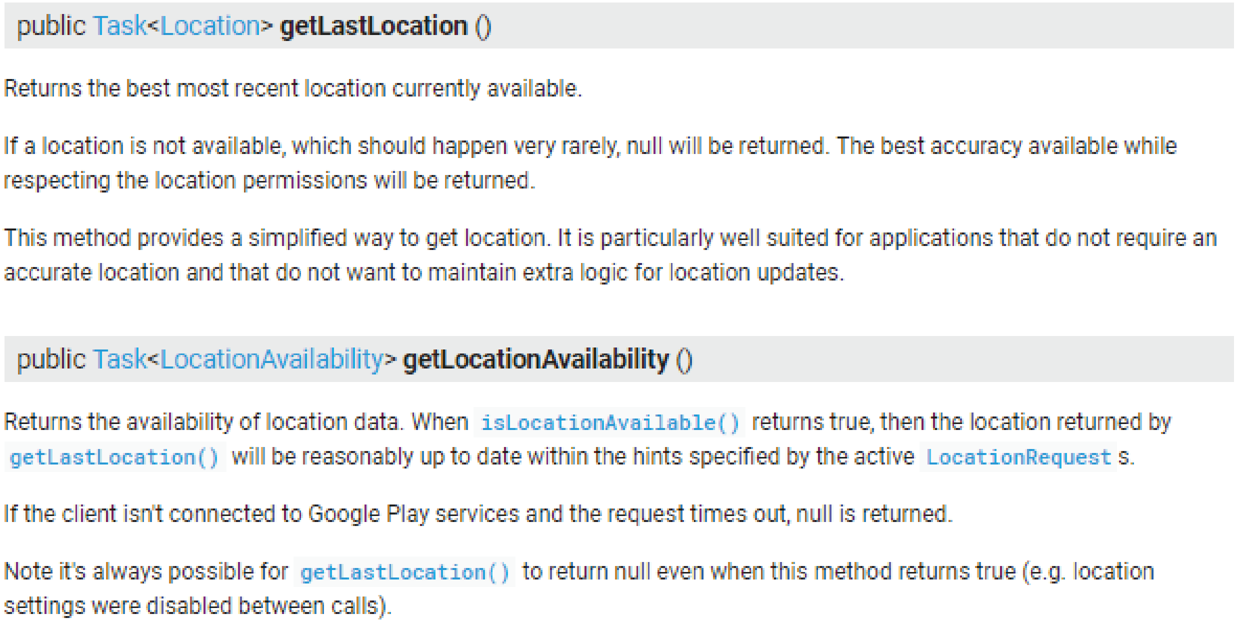
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public Task<Location> **getLastLocation** () Returns the best most recent location currently available. If a location is not available, which should happen very rarely, null will be returned. The best accuracy available while respecting the location permissions will be returned. This method provides a simplified way to get location. It is particularly well suited for applications that do not require an accurate location and that do not want to maintain extra logic for location updates. public Task<LocationAvailability> **getLocationAvailability** () Returns the availability of location data. When `isLocationAvailable()` returns true, then the location returned by `getLastLocation()` will be reasonably up to date within the hints specified by the active `LocationRequest` s. If the client isn't connected to Google Play services and the request times out, null is returned. Note it's always possible for `getLastLocation()` to return null even when this method returns true (e.g. location settings were disabled between calls). https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

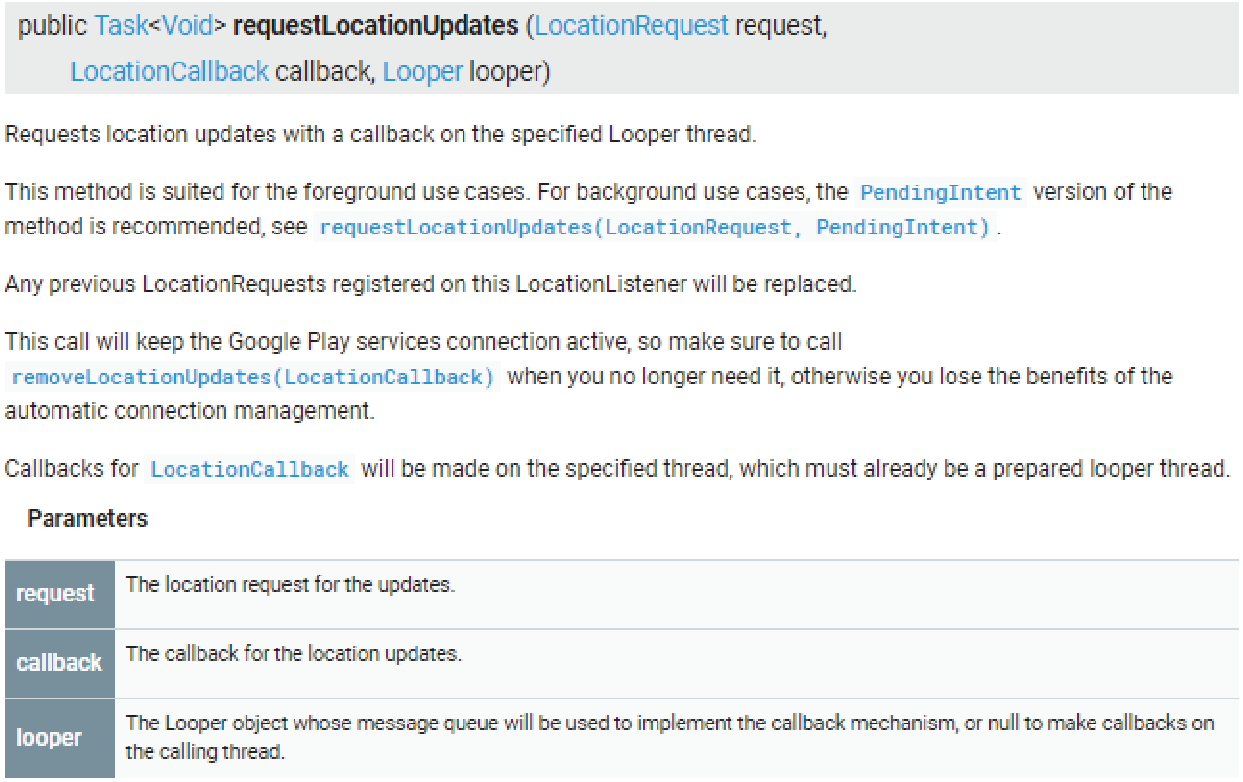
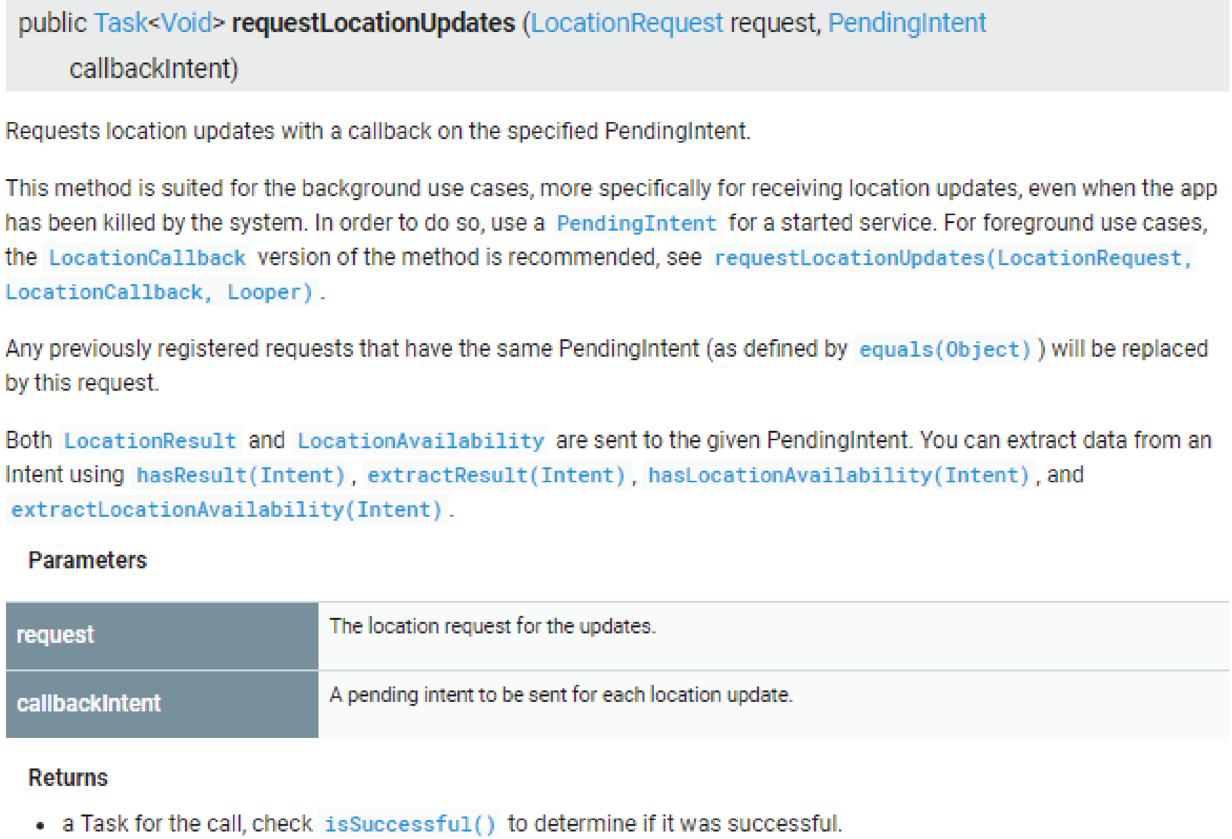
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public Task<Void> **requestLocationUpdates** (LocationRequest request, LocationCallback callback, Looper looper) Requests location updates with a callback on the specified Looper thread. This method is suited for the foreground use cases. For background use cases, the `PendingIntent` version of the method is recommended, see `requestLocationUpdates(LocationRequest, PendingIntent)`. Any previous LocationRequests registered on this LocationListener will be replaced. This call will keep the Google Play services connection active, so make sure to call `removeLocationUpdates(LocationCallback)` when you no longer need it, otherwise you lose the benefits of the automatic connection management. Callbacks for `LocationCallback` will be made on the specified thread, which must already be a prepared looper thread. **Parameters** **request** The location request for the updates. **callback** The callback for the location updates. **looper** The Looper object whose message queue will be used to implement the callback mechanism, or null to make callbacks on the calling thread. https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>


public Task<Void> **requestLocationUpdates** (LocationRequest request, PendingIntent callbackIntent)

Requests location updates with a callback on the specified PendingIntent.

This method is suited for the background use cases, more specifically for receiving location updates, even when the app has been killed by the system. In order to do so, use a `PendingIntent` for a started service. For foreground use cases, the `LocationCallback` version of the method is recommended, see `requestLocationUpdates(LocationRequest, LocationCallback, Looper)` .

Any previously registered requests that have the same PendingIntent (as defined by `equals(Object)`) will be replaced by this request.

Both `LocationResult` and `LocationAvailability` are sent to the given PendingIntent. You can extract data from an Intent using `hasResult(Intent)` , `extractResult(Intent)` , `hasLocationAvailability(Intent)` , and `extractLocationAvailability(Intent)` .

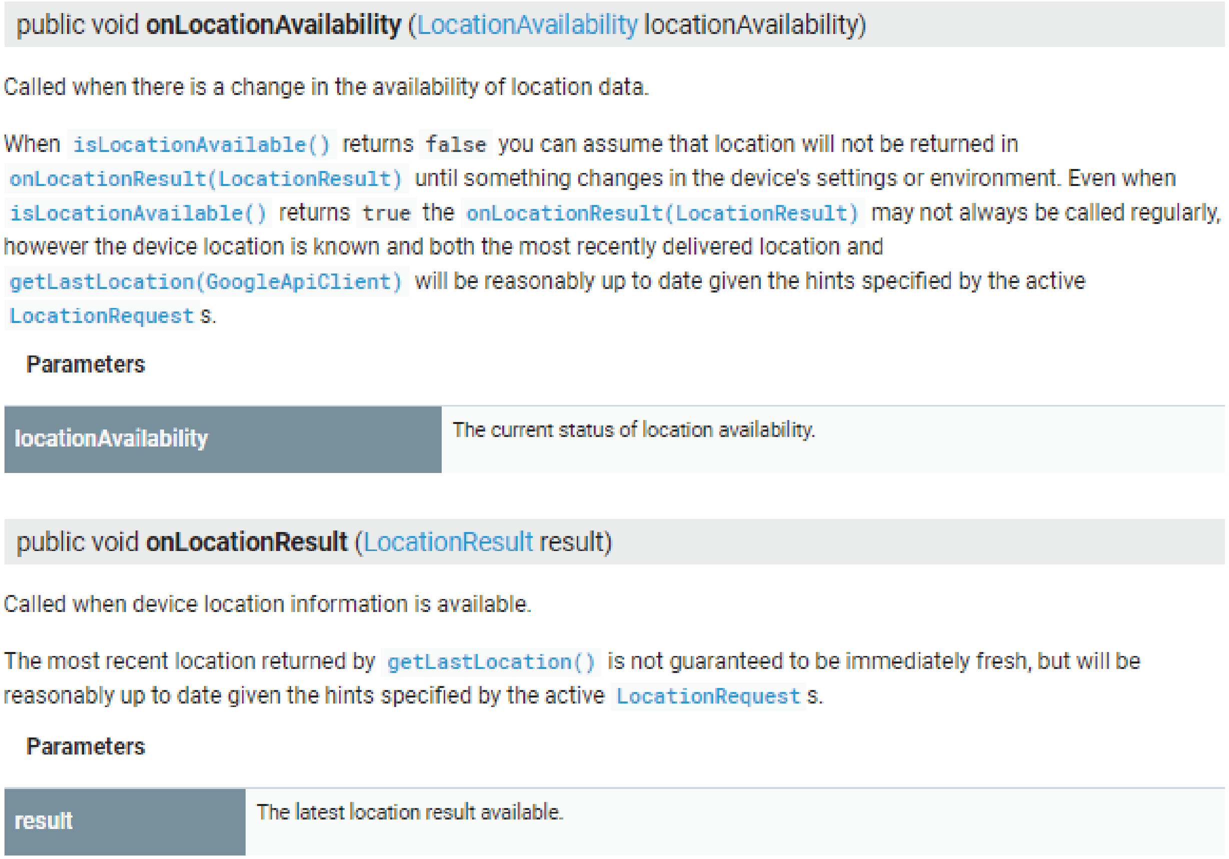
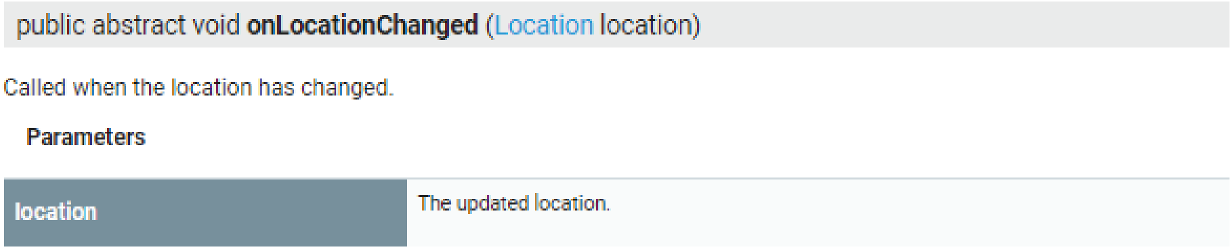
Parameters

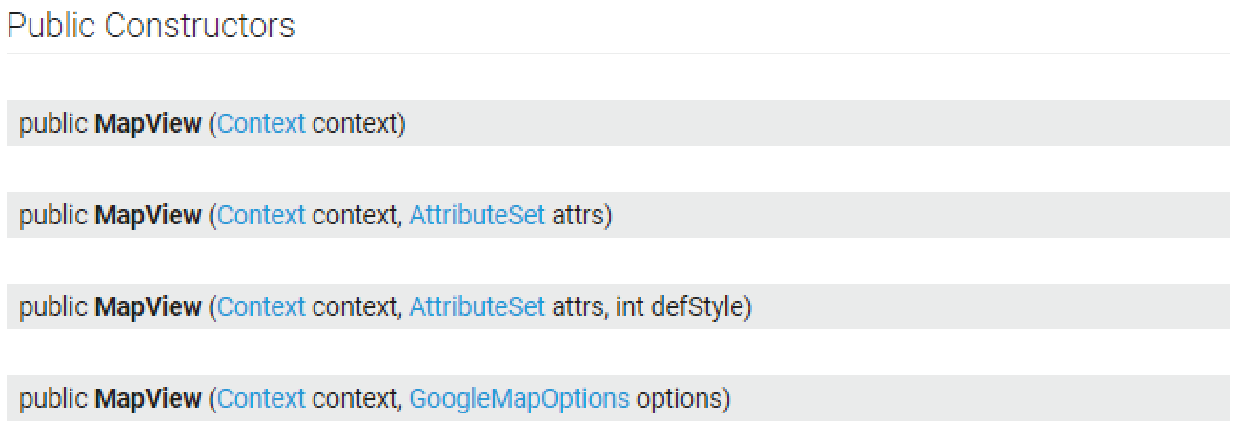
<table>
<tr><td><b>request</b></td><td>The location request for the updates.</td></tr>
<tr><td><b>callbackIntent</b></td><td>A pending intent to be sent for each location update.</td></tr>
</table>

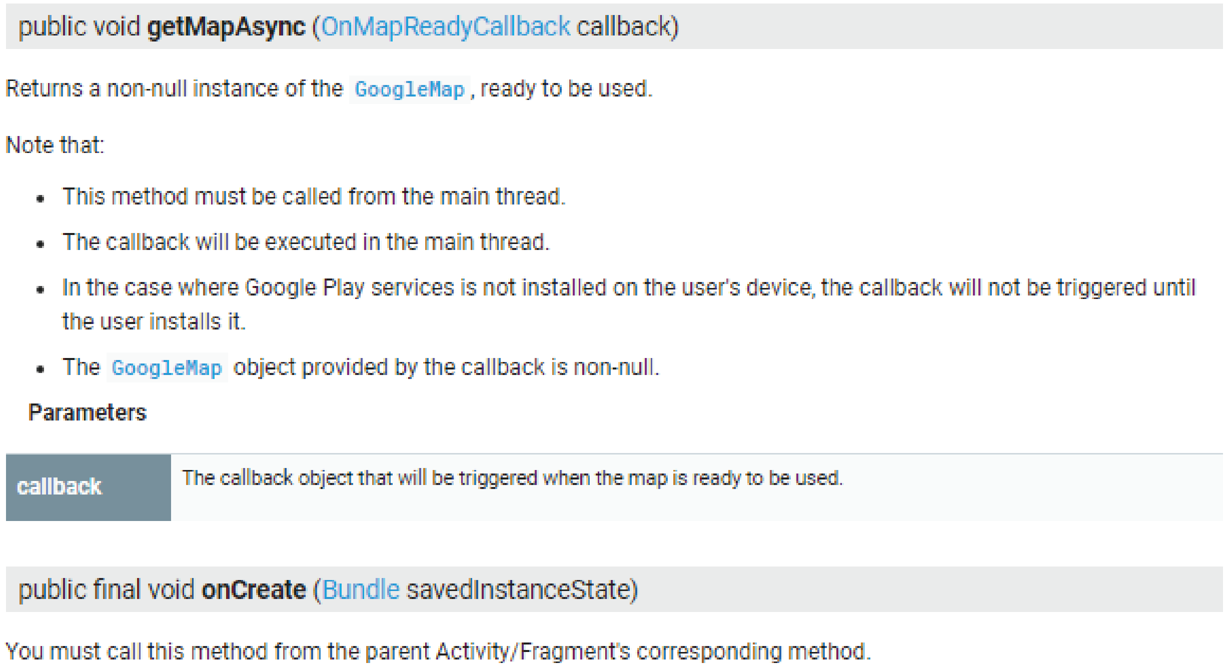
Returns

- a Task for the call, check `isSuccessful()` to determine if it was successful.

https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient
</td></tr>
</table>

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public void **onLocationAvailability** (LocationAvailability locationAvailability) Called when there is a change in the availability of location data. When `isLocationAvailable()` returns `false` you can assume that location will not be returned in `onLocationResult(LocationResult)` until something changes in the device's settings or environment. Even when `isLocationAvailable()` returns `true` the `onLocationResult(LocationResult)` may not always be called regularly, however the device location is known and both the most recently delivered location and `getLastLocation(GoogleApiClient)` will be reasonably up to date given the hints specified by the active `LocationRequest` s. **Parameters** **locationAvailability** The current status of location availability. public void **onLocationResult** (LocationResult result) Called when device location information is available. The most recent location returned by `getLastLocation()` is not guaranteed to be immediately fresh, but will be reasonably up to date given the hints specified by the active `LocationRequest` s. **Parameters** **result** The latest location result available. https://developers.google.com/android/reference/com/google/android/gms/location/LocationCallback  public abstract void **onLocationChanged** (Location location) Called when the location has changed. **Parameters** **location** The updated location.

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	https://developers.google.com/android/reference/com/google/android/gms/location/LocationListener  Public Constructors public **MapView** (Context context) public **MapView** (Context context, AttributeSet attrs) public **MapView** (Context context, AttributeSet attrs, int defStyle) public **MapView** (Context context, GoogleMapOptions options) https://developers.google.com/android/reference/com/google/android/gms/maps/MapView

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public void **getMapAsync** (OnMapReadyCallback callback) Returns a non-null instance of the `GoogleMap`, ready to be used. Note that: - This method must be called from the main thread. - The callback will be executed in the main thread. - In the case where Google Play services is not installed on the user's device, the callback will not be triggered until the user installs it. - The `GoogleMap` object provided by the callback is non-null. **Parameters** callback: The callback object that will be triggered when the map is ready to be used. public final void **onCreate** (Bundle savedInstanceState) You must call this method from the parent Activity/Fragment's corresponding method. https://developers.google.com/android/reference/com/google/android/gms/maps/MapView
[1D] presenting, via an interactive display of the first device, a first interactive, georeferenced map and a first set of one or more user-selectable symbols corresponding to a first set of one or more of the second devices, wherein the first set of symbols are positioned on the	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of: presenting, via an interactive display of the first device, a first interactive, georeferenced map and a first set of one or more user-selectable symbols corresponding to a first set of one or more of the second devices, wherein the first set of symbols are positioned on the first georeferenced map at respective positions corresponding to the locations of the first set of second devices, and wherein first georeferenced map data relate positions on the first georeferenced map to spatial coordinates. **Regarding Find My Device** and Android Device Manager, the Accused Products display, to the user on the display of the first device, a map with one or more symbols corresponding to one or more second devices. The map is interactive because the user may control the display of the map, e.g., pan, zoom, and/or effect change to the map in an otherwise interactive manner. The map is georeferenced for at least the reason that one or more symbols are associated with spatial locations, i.e., coordinates. The symbols

US9467838	Exemplary Supporting Evidence Regarding Accused Products
first georeferenced map at respective positions corresponding to the locations of the first set of second devices, and wherein first georeferenced map data relate positions on the first georeferenced map to spatial coordinates;	are user-selectable because a user may touch the display to select the device associated with the symbol. For example, the user may input a touch selection directly on or near the portion of the display corresponding to the symbol's coordinates on the map to effect a selection of the device. **<u>Regarding Google Maps</u>**, Google Latitude, Google Plus, Google Hangouts, Google Messages, Google Allo, Google Duo, Google Chrome, and Android Messenger, the Accused Products display, to the user on the display of the first device, a map with one or more symbols corresponding to one or more second users (or second devices corresponding to the second users). The map is interactive because the user may control the display of the map, e.g., pan, zoom, and/or effect change to the map in an otherwise interactive manner. The map is georeferenced for at least the reason that one or more symbols are associated with spatial locations, i.e., coordinates. The symbols are user-selectable because a user may touch the display to select the user or device associated with the symbol. For example, the user may input a touch selection directly on or near the portion of the display corresponding to the symbol's coordinates on the map to effect a selection of the user or device. **<u>Exemplary Support for Find My Device</u>**:

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

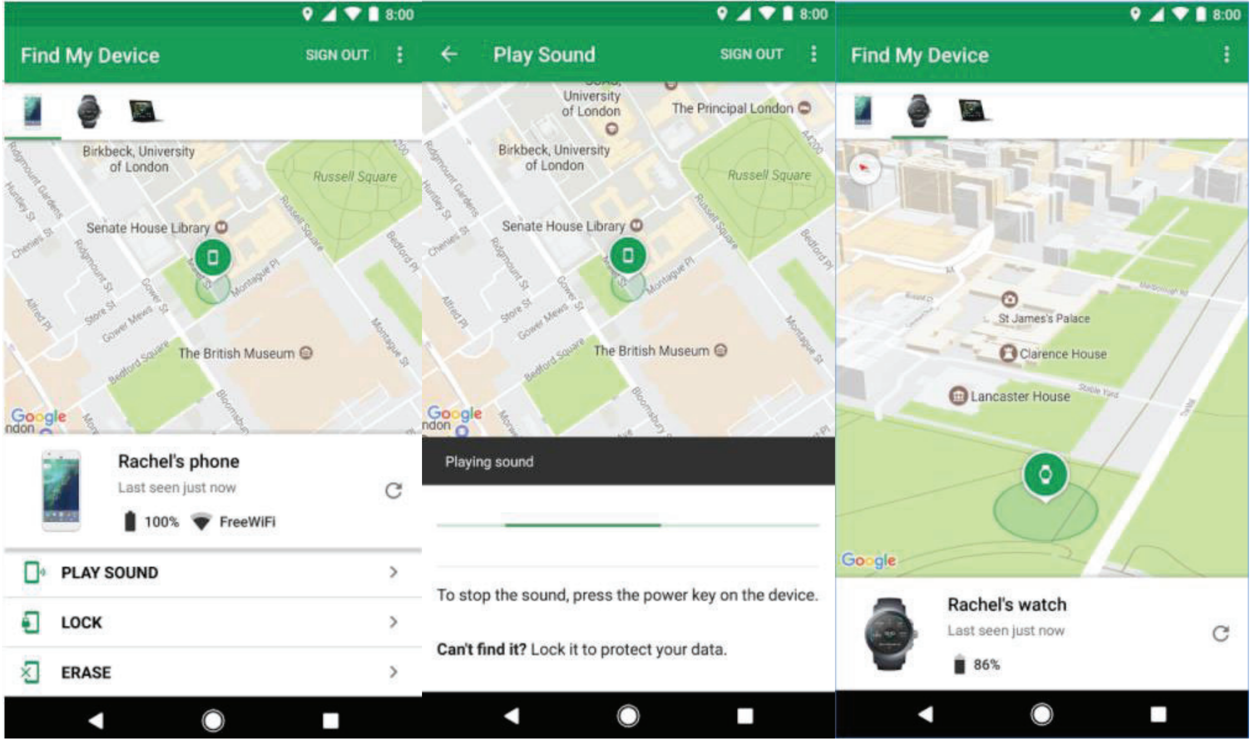
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 8:00 Find My Device SIGN OUT Birkbeck, University of London Russell Square Senate House Library The British Museum Google Rachel's phone Last seen just now 100% FreeWiFi PLAY SOUND LOCK ERASE Play Sound University of London The Principal London Playing sound St James's Palace Clarence House Lancaster House Rachel's watch 86% To stop the sound, press the power key on the device. **Can't find it?** Lock it to protect your data. https://play.google.com/store/apps/details?id=com.google.android.apps.adm&hl=en

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

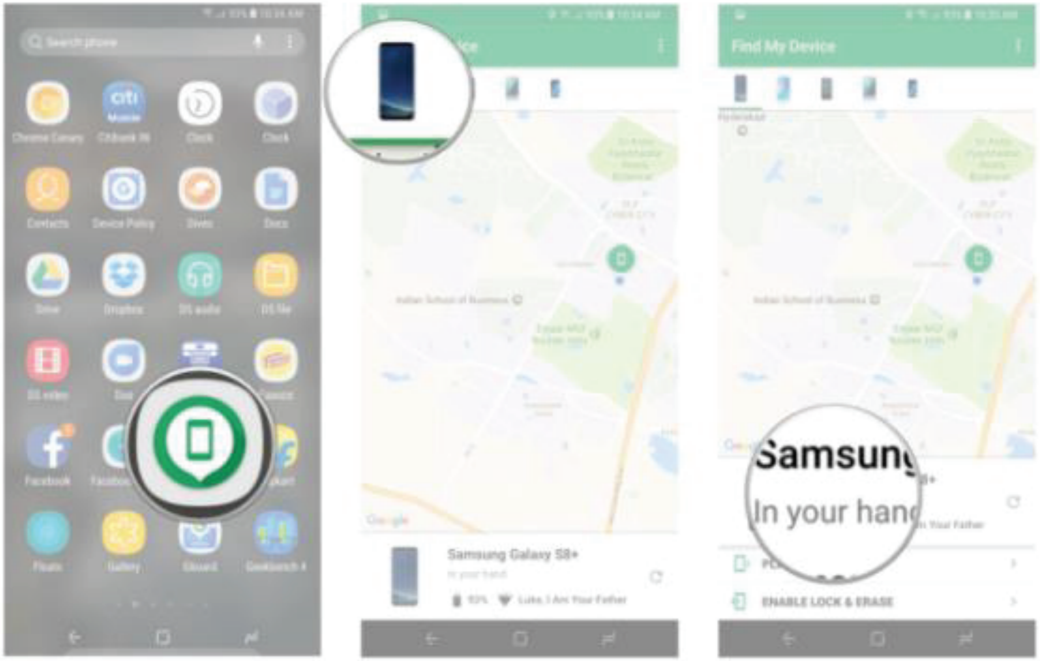
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	Once you're signed in to Find My Device, you'll see a map with your current location as well as the make and model of your phone, and two options — Play Sound, and Enable Lock & Erase. Hitting the latter option will allow you to start using the Lock and Erase functions. If you've signed into more than one phone, you can select a particular device by browsing the list at the top of the screen. 1. Open **Find My Device** from your home screen or app drawer. 2. Select your phone from the **list of devices at the top of the screen**. 3. See if your phone is **discoverable**.  Find My Device Samsung Galaxy S8+ Samsung In your hand ENABLE LOCK & ERASE https://www.androidcentral.com/find-my-device

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

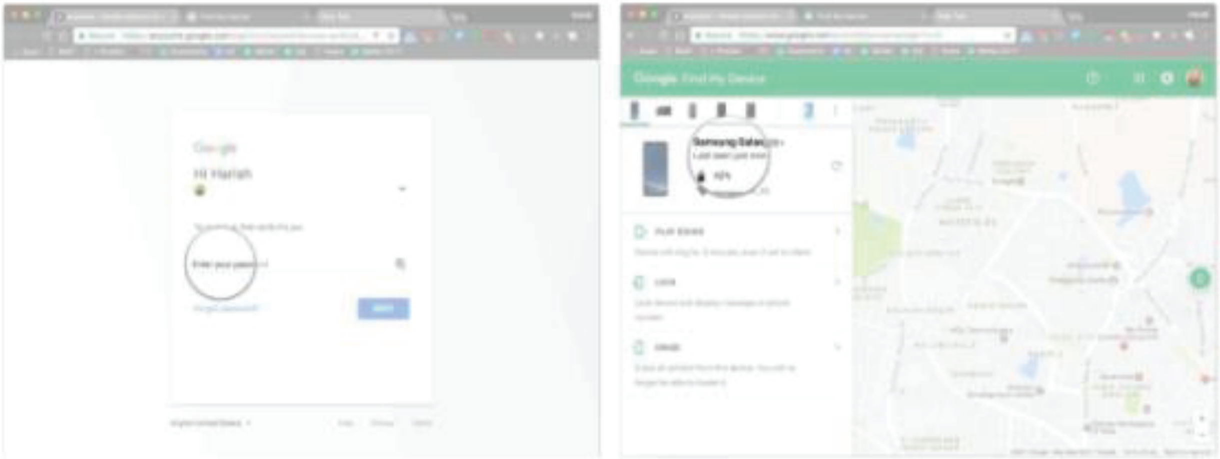
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to locate your phone over the internet If you've lost your phone, you can remotely locate it through the Find My Device website. You'll need to sign in to the Google account that was used to set up Find My Device. It takes a few seconds, but the service should be able to track your phone. Alternatively, you can also do a Google search for "find my phone" to locate your handset. 1. Head to the **Find My Device website**. 2. Sign in to your **Google account**. 3. Check if your device is **visible**.  Google Hi Hannah Google Find My Device Samsung Galaxy https://www.androidcentral.com/find-my-device

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	<u>Exemplary Support for Google Maps</u>:

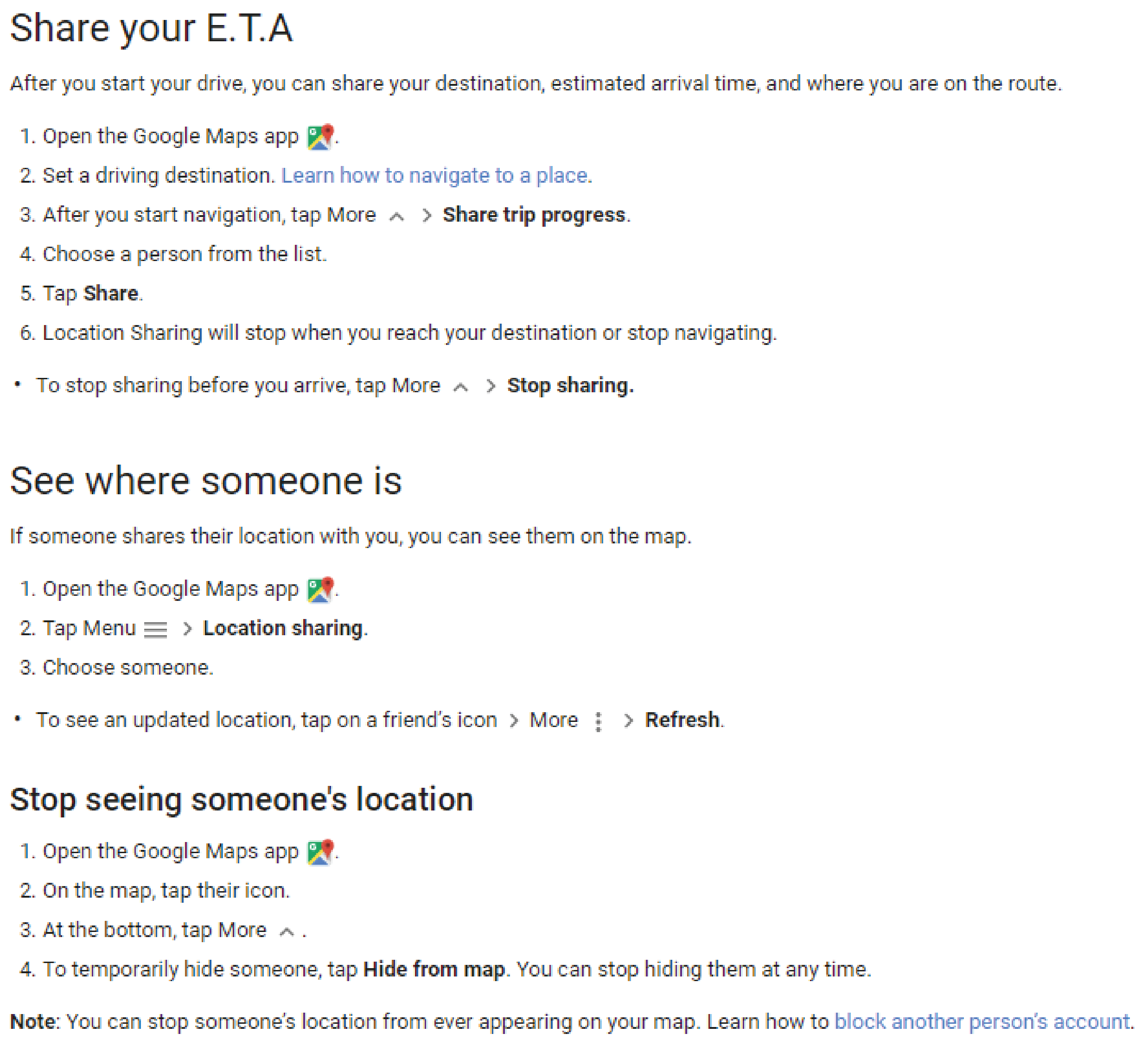
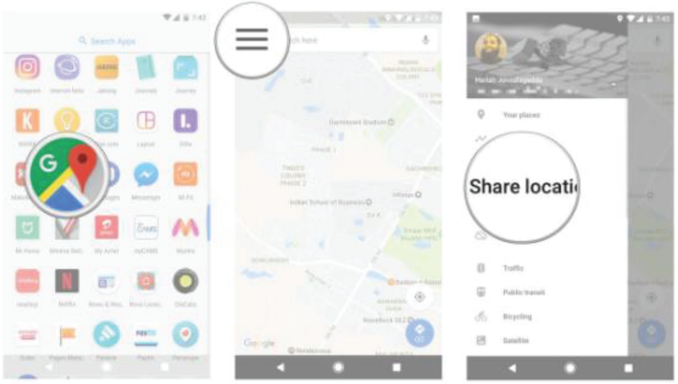
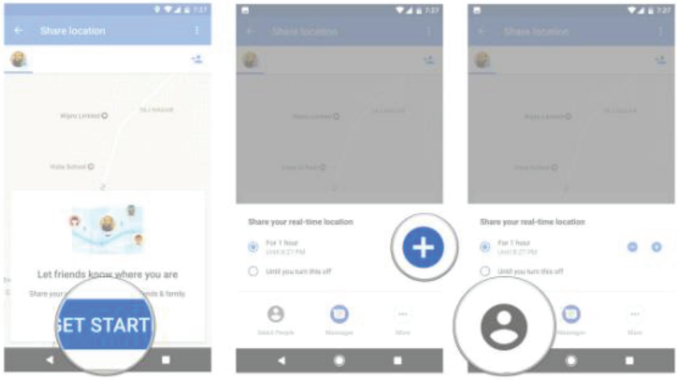
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 ## Share your E.T.A After you start your drive, you can share your destination, estimated arrival time, and where you are on the route. 1. Open the Google Maps app. 2. Set a driving destination. Learn how to navigate to a place. 3. After you start navigation, tap More ^ > **Share trip progress**. 4. Choose a person from the list. 5. Tap **Share**. 6. Location Sharing will stop when you reach your destination or stop navigating. • To stop sharing before you arrive, tap More ^ > **Stop sharing.** ## See where someone is If someone shares their location with you, you can see them on the map. 1. Open the Google Maps app. 2. Tap Menu ☰ > **Location sharing**. 3. Choose someone. • To see an updated location, tap on a friend's icon > More ⋮ > **Refresh**. ## Stop seeing someone's location 1. Open the Google Maps app. 2. On the map, tap their icon. 3. At the bottom, tap More ^ . 4. To temporarily hide someone, tap **Hide from map**. You can stop hiding them at any time. **Note**: You can stop someone's location from ever appearing on your map. Learn how to block another person's account. https://support.google.com/maps/answer/7326816?co=GENIE.Platform%3DAndroid&oco=1

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to share your location in Google Maps 1. Open **Google Maps** from the app drawer or the home screen. 2. Tap the **hamburger menu** (the three horizontal lines) on the top left corner of the screen. 3. Select **Share location**.  4. Tap **Get Started**. 5. Use the **+** icon to select a time period or select the **Until you turn this off** setting to share your location indefinitely. 6. Tap **Select People**.  Share location Let friends know where you are ET START Share your real-time location For 1 hour Until you turn this off https://www.androidcentral.com/how-share-location-google-maps

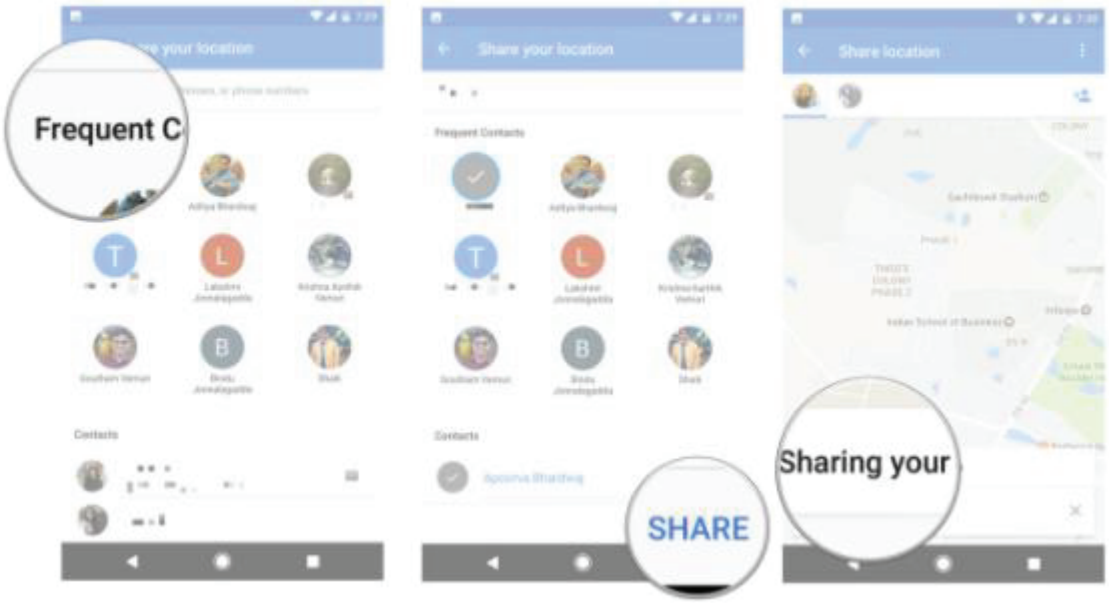
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	7. You'll see a list of your frequent contacts at the top, along with a full list of contacts. **Pick the contacts** by tapping their name. 8. Once you've selected the contacts you want to share your location to, tap **Share**. 9. You'll see a message saying that the selected contact can view your location.  Frequent C Share your location Frequent Contacts Contacts SHARE Share location Sharing your https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

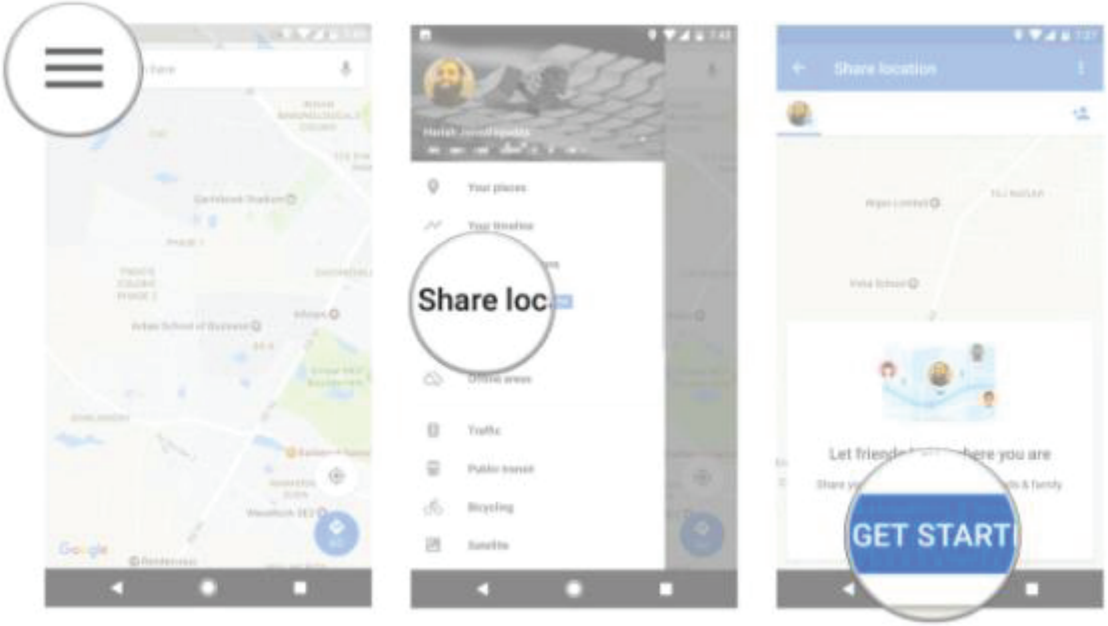
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to create a shareable link You can also create a link and use it to share your location easily. Here's how to do it: 1. Tap the **hamburger menu** on the top left corner of the screen. 2. Select **Share location**. 3. Tap **Get Started**.  Share loc Share location Let friends ... where you are GET START https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

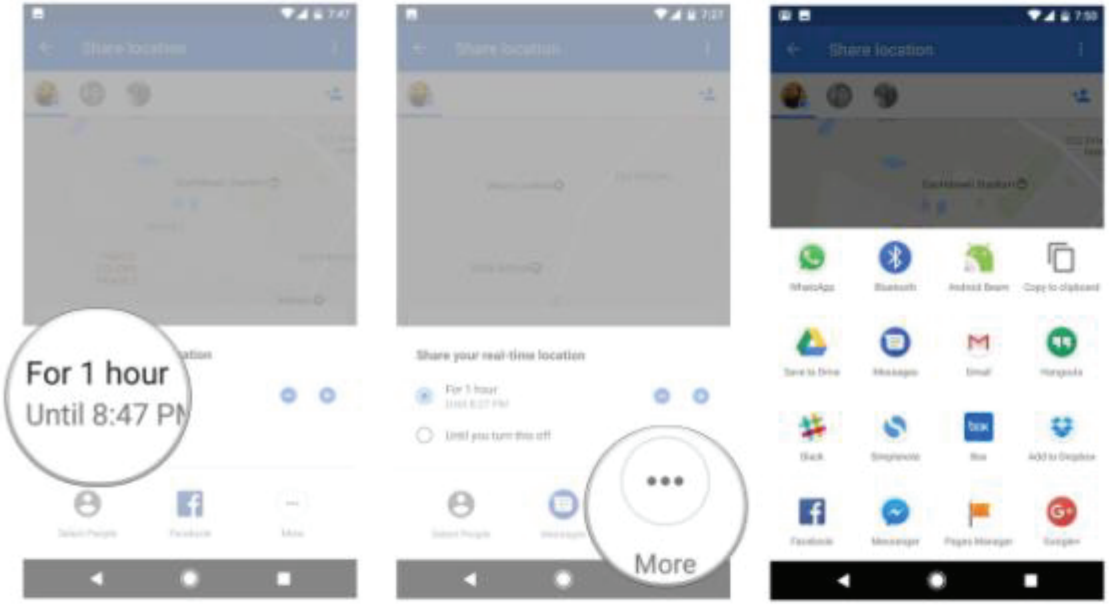
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	4. Select the **amount of time** you want to share your location. 5. Tap **More**. 6. Select **your app of choice** to create and send a unique URL that broadcasts your current location. You can email it, send the link via Messenger, or even tweet it to the intended recipient.  Share location For 1 hour Until 8:47 PM Facebook More Share location Share your real-time location For 1 hour Until you turn this off More Share location WhatsApp Bluetooth Android Beam Copy to clipboard Save to Drive Messages Gmail Hangouts Slack Simplenote Box Add to Dropbox Facebook Messenger Pages Manager Google+ https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

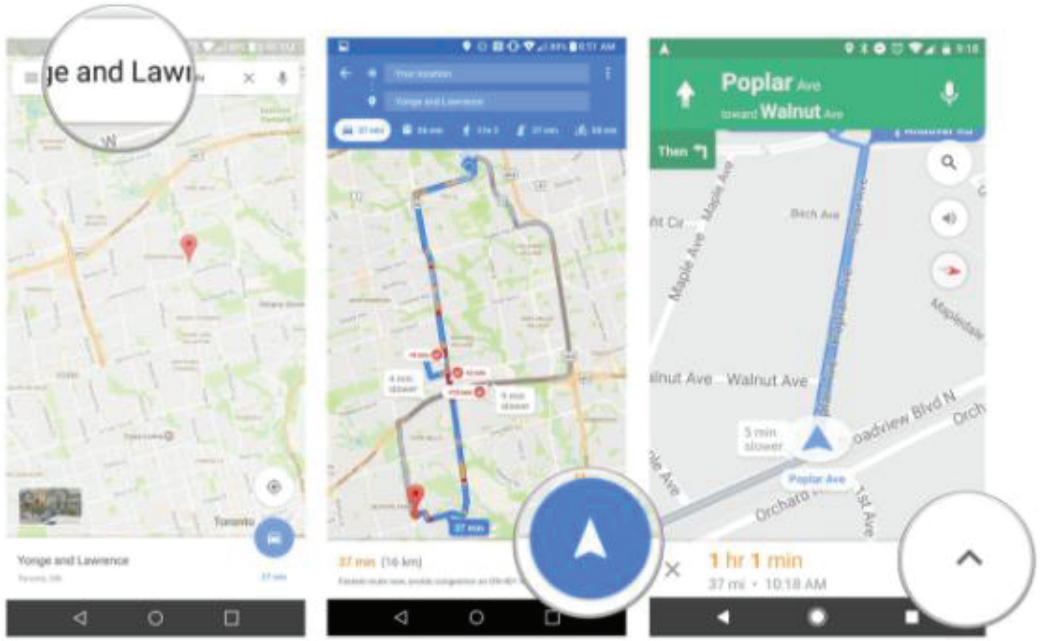
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to share your navigation directions while you walk, drive or transit One of the best ways to share your location with someone is during a drive, walk or transit. If you're meeting a friend or family member somewhere, or navigating towards their home, Google Maps lets you share your location with them for the duration of the trip. It's magic! 1. In the **search bar** enter your destination. 2. Pick your navigation type (drive, transit, walk) and press the blue **navigate button**. 3. Tap the arrow next to the **time-to-destination** number at the bottom of the screen.  ge and Lawr Yonge and Lawrence Toronto 37 min (16 km) 37 min Poplar Ave toward Walnut Ave Then Maple Ave Birch Ave Walnut Ave 5 min slower Poplar Ave Orchard 1st Ave 1 hr 1 min 37 mi • 10:18 AM https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

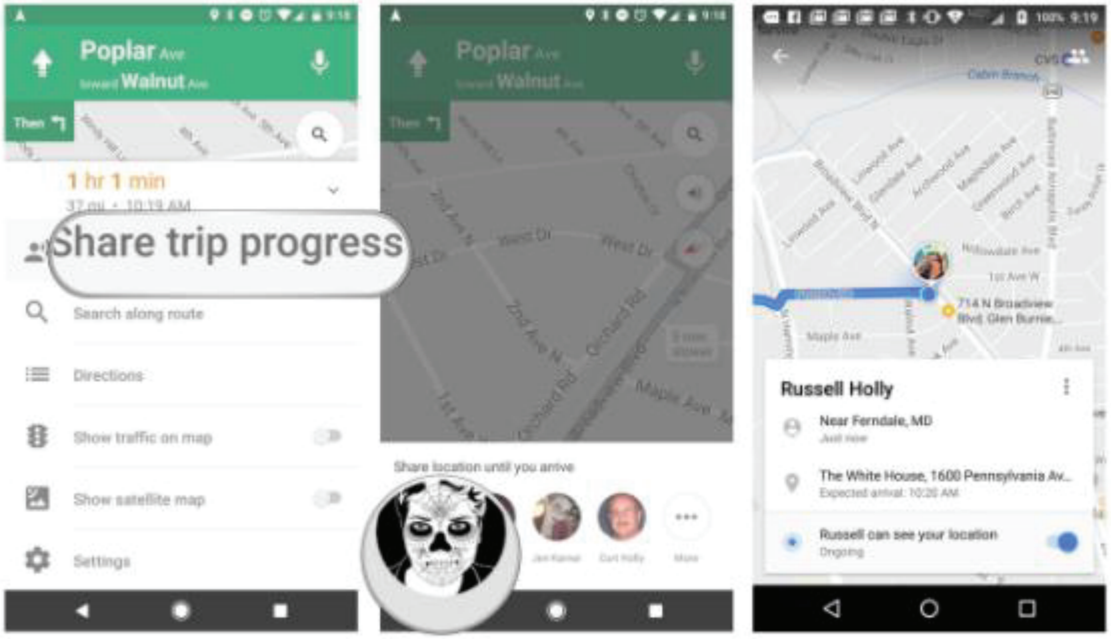
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	4. Tap **Share trip progress**. 5. Choose one or more contacts to share trip progress.  Poplar Ave toward Walnut Ave Then 1 hr 1 min 37 mi · 10:19 AM Share trip progress Search along route Directions Show traffic on map Show satellite map Settings Share location until you arrive Jen Karner Carl Holly More Russell Holly Near Ferndale, MD Just now The White House, 1600 Pennsylvania Av... Expected arrival: 10:20 AM Russell can see your location Ongoing 714 N Broadview Blvd, Glen Burnie... Cabin Branch CVS You can also stop sharing your location with someone before a trip ends. https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

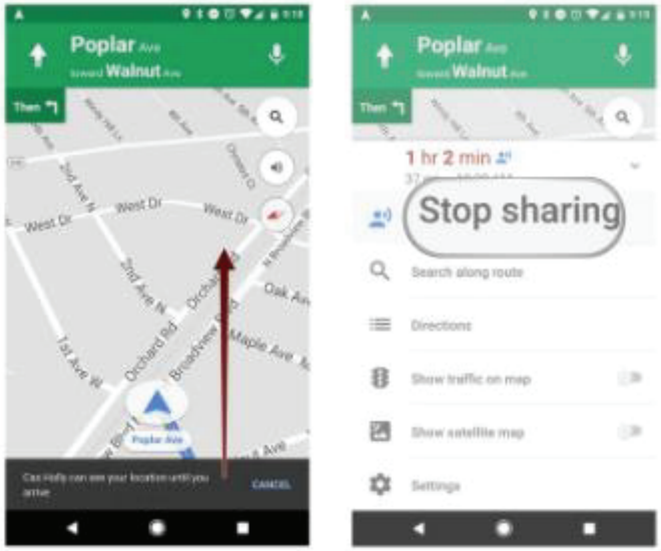
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	1. Tap the arrow next to the **time-to-destination** number at the bottom of the screen. 2. Tap **Stop sharing**.  Poplar Ave Walnut Ave Then West Dr 2nd Ave N Orchard Rd Broadview Oak Ave Maple Ave 1st Ave W Poplar Ave CANCEL 1 hr 2 min Stop sharing Search along route Directions Show traffic on map Show satellite map Settings That's it! Are you excited that location sharing is back in Google Maps? How often do you use the feature? https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

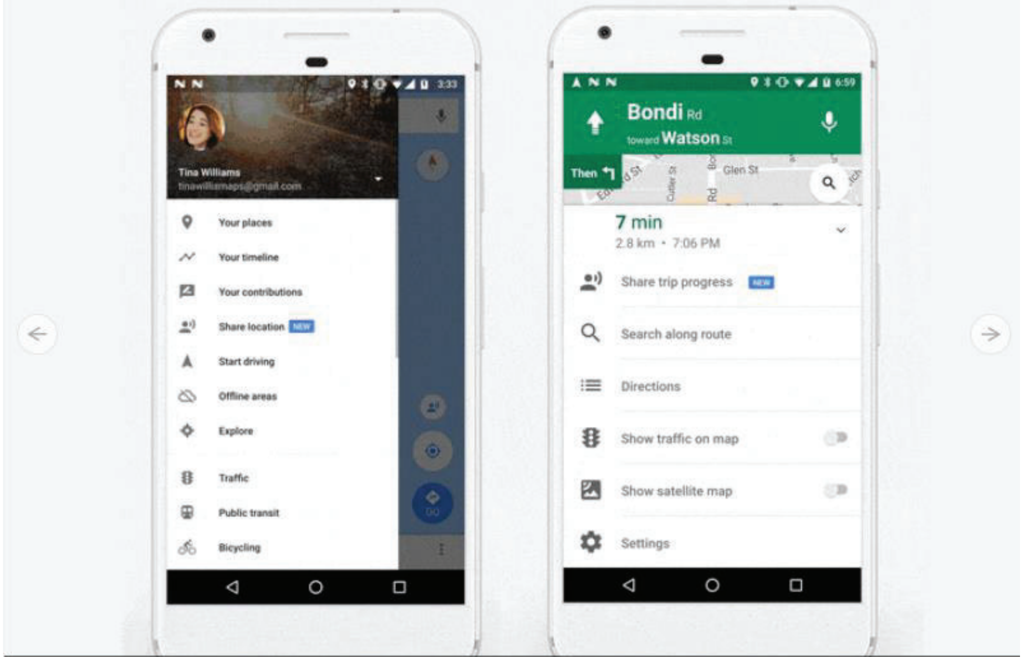
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Tina Williams tinawilliamsmaps@gmail.com Your places Your timeline Your contributions Share location NEW Start driving Offline areas Explore Traffic Public transit Bicycling Bondi Rd toward Watson St Then Glen St 7 min 2.8 km • 7:06 PM Share trip progress NEW Search along route Directions Show traffic on map Show satellite map Settings Ways to start location sharing: (left) from the app navigation drawer, and (right) from the menu, you can drag up while navigating somewhere. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

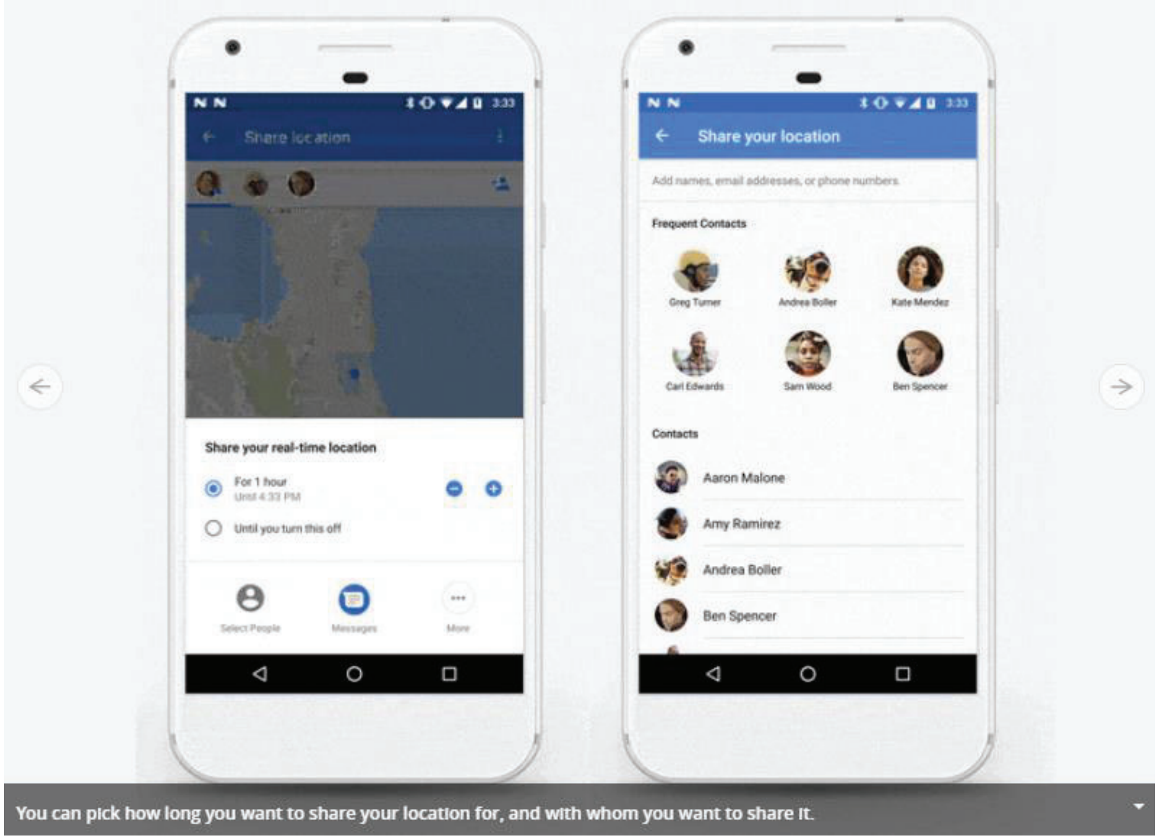
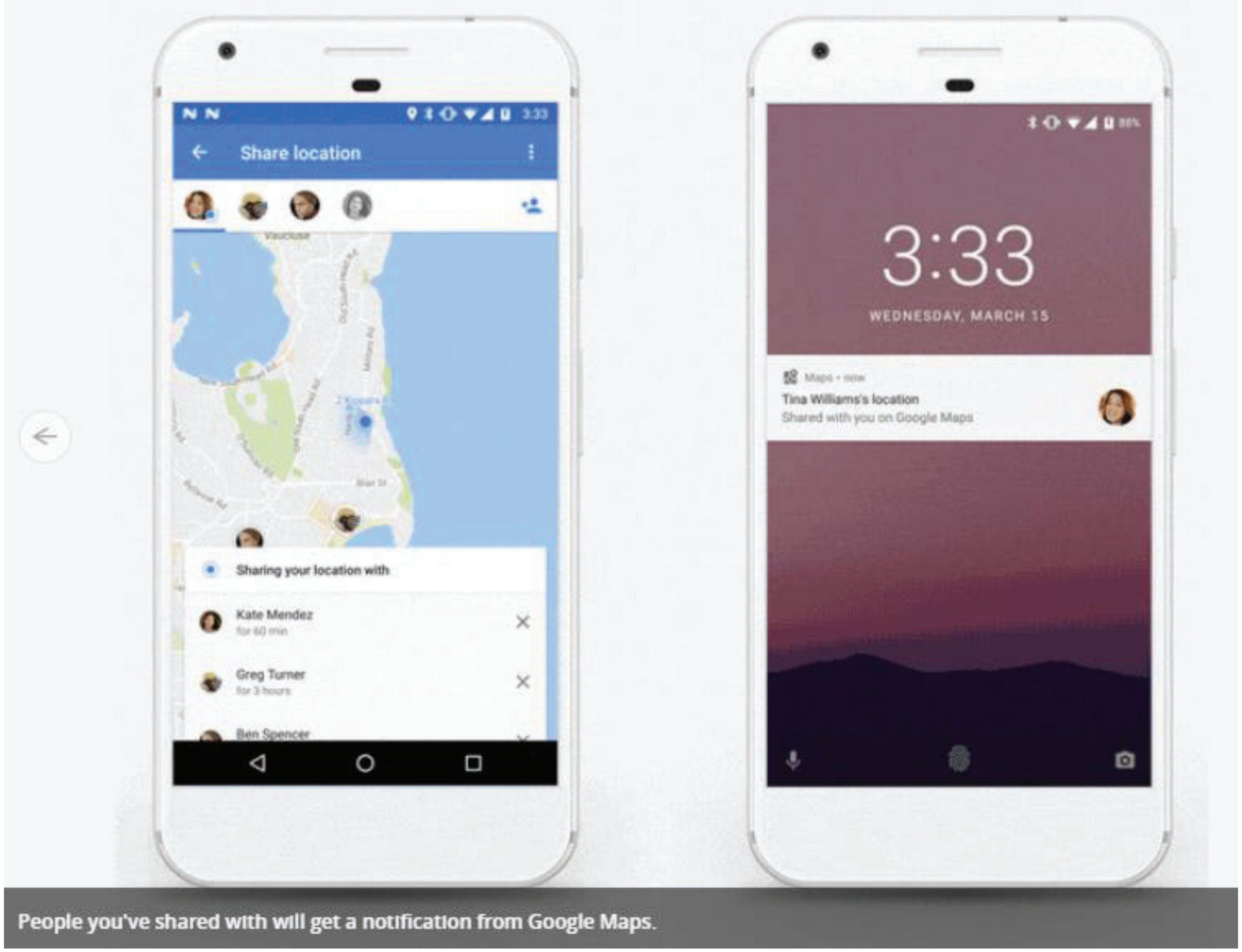
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Share location Share your real-time location For 1 hour Until 4:33 PM Until you turn this off Select People Messages More Share your location Add names, email addresses, or phone numbers. Frequent Contacts Greg Turner Andrea Boller Kate Mendez Carl Edwards Sam Wood Ben Spencer Contacts Aaron Malone Amy Ramirez Andrea Boller Ben Spencer You can pick how long you want to share your location for, and with whom you want to share it. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Share location Vaucluse Sharing your location with Kate Mendez for 60 min Greg Turner for 3 hours Ben Spencer 3:33 WEDNESDAY, MARCH 15 Maps • now Tina Williams's location Shared with you on Google Maps People you've shared with will get a notification from Google Maps. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

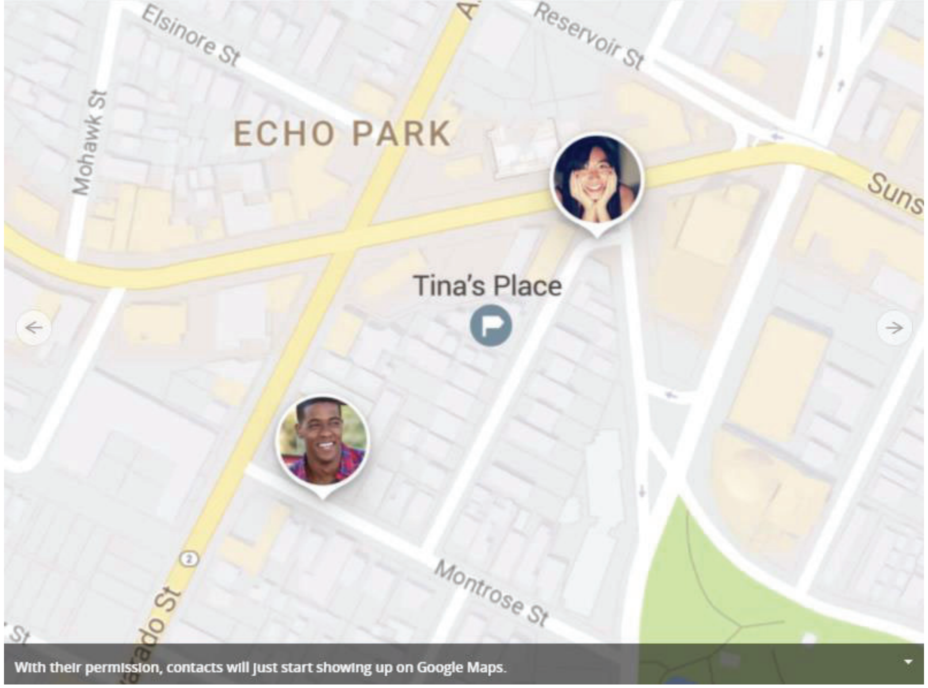
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Elsinore St Reservoir St Mohawk St ECHO PARK Suns Tina's Place Montrose St ado St With their permission, contacts will just start showing up on Google Maps. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/ **<u>Exemplary Find My Device Screenshots</u>**:

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

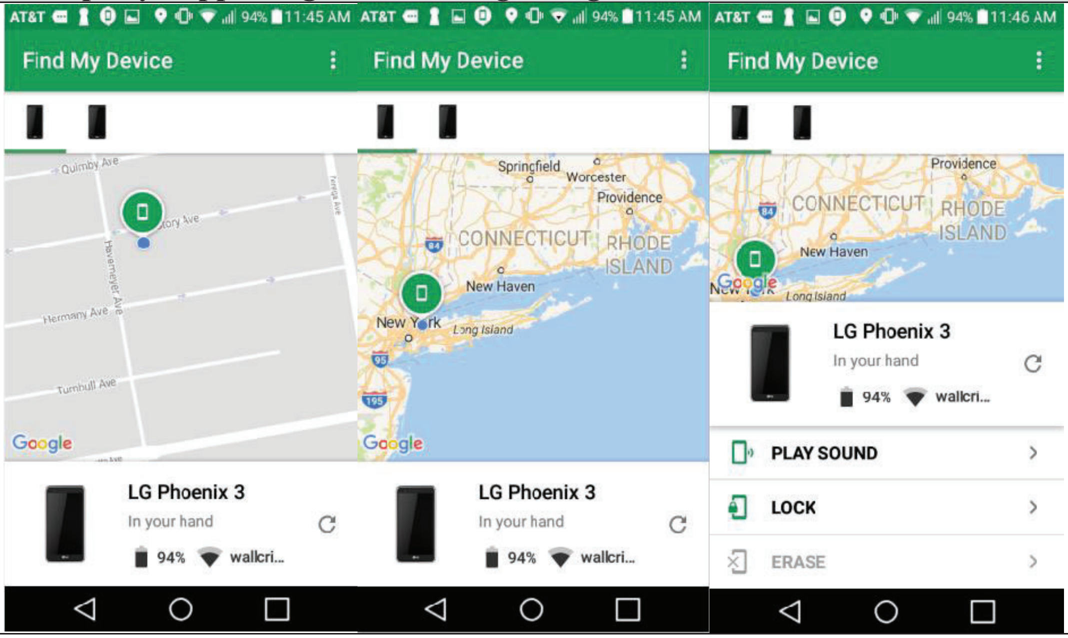
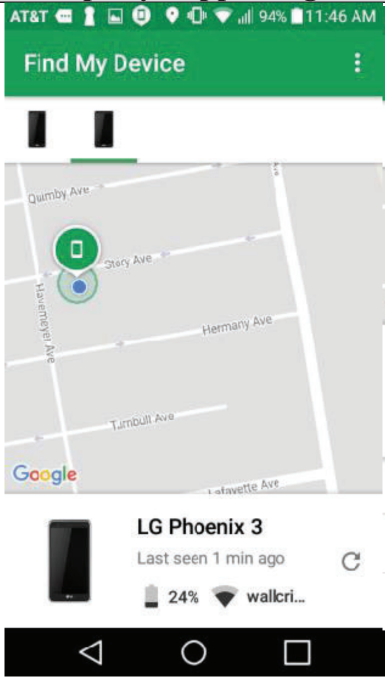
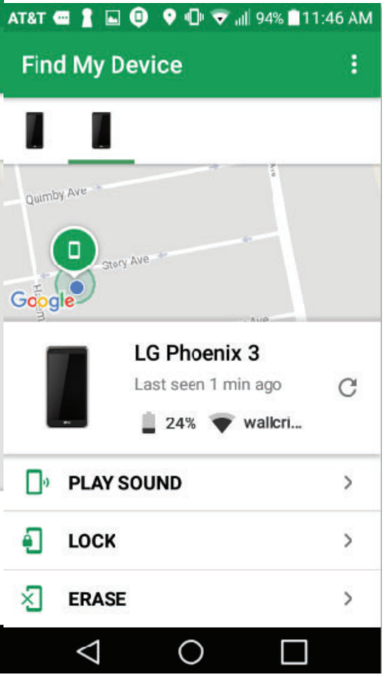
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 94% 11:45 AM Find My Device Quimby Ave Havemeyer Ave Hermany Ave Turnbull Ave Google LG Phoenix 3 In your hand 94% wallcri... AT&T 94% 11:45 AM Find My Device Springfield Worcester Providence CONNECTICUT RHODE ISLAND New Haven New York Long Island Google LG Phoenix 3 In your hand 94% wallcri... AT&T 94% 11:46 AM Find My Device Providence CONNECTICUT RHODE ISLAND New Haven Long Island Google LG Phoenix 3 In your hand 94% wallcri... PLAY SOUND LOCK ERASE

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 94% 11:46 AM Find My Device Quimby Ave Story Ave Havemeyer Ave Hermany Ave Turnbull Ave Lafayette Ave Google LG Phoenix 3 Last seen 1 min ago 24% wallcri...  AT&T 94% 11:46 AM Find My Device Quimby Ave Story Ave Google LG Phoenix 3 Last seen 1 min ago 24% wallcri... PLAY SOUND LOCK ERASE

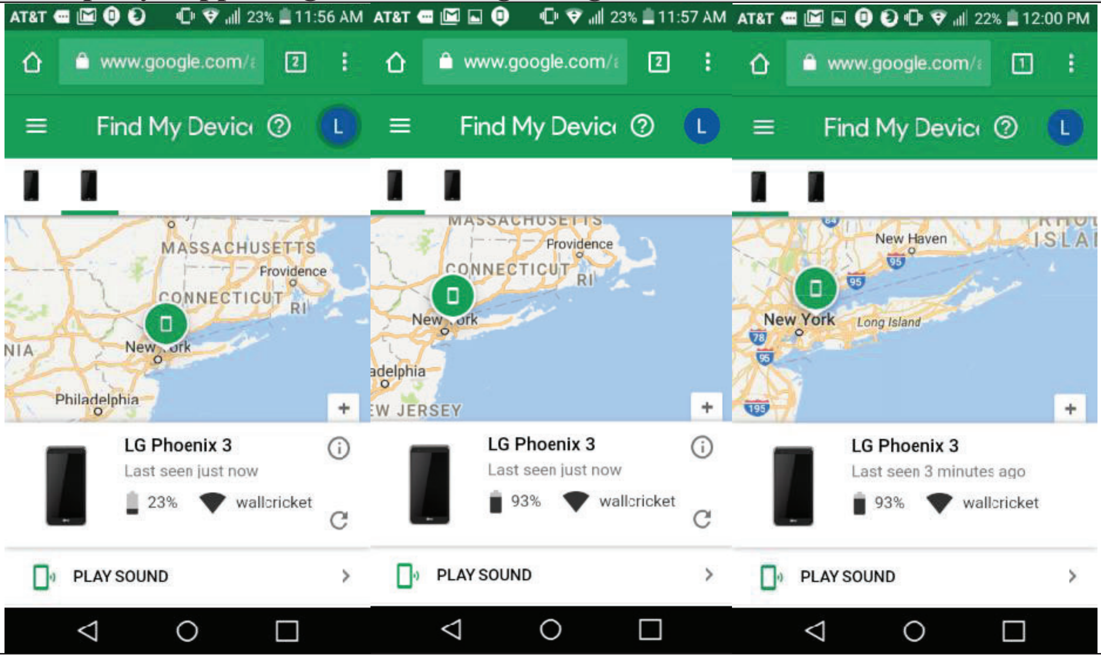
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 23% 11:56 AM www.google.com/ Find My Devic L MASSACHUSETTS Providence CONNECTICUT RI NIA New York Philadelphia LG Phoenix 3 Last seen just now 23% wallcricket PLAY SOUND AT&T 23% 11:57 AM www.google.com/ Find My Devic MASSACHUSETTS Providence CONNECTICUT New York adelphia W JERSEY LG Phoenix 3 Last seen just now 93% wallcricket PLAY SOUND AT&T 22% 12:00 PM www.google.com/ Find My Devic New Haven ISLA New York Long Island LG Phoenix 3 Last seen 3 minutes ago 93% wallcricket PLAY SOUND

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

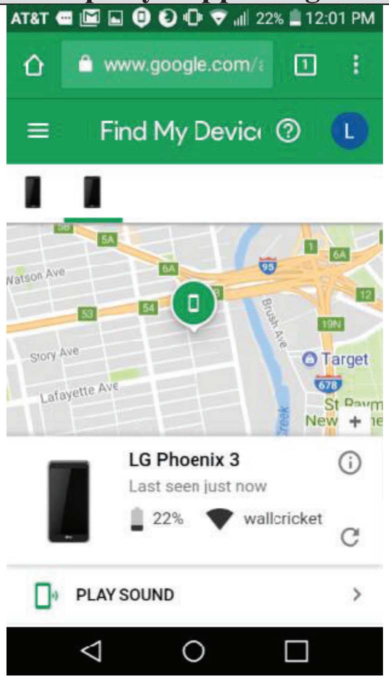
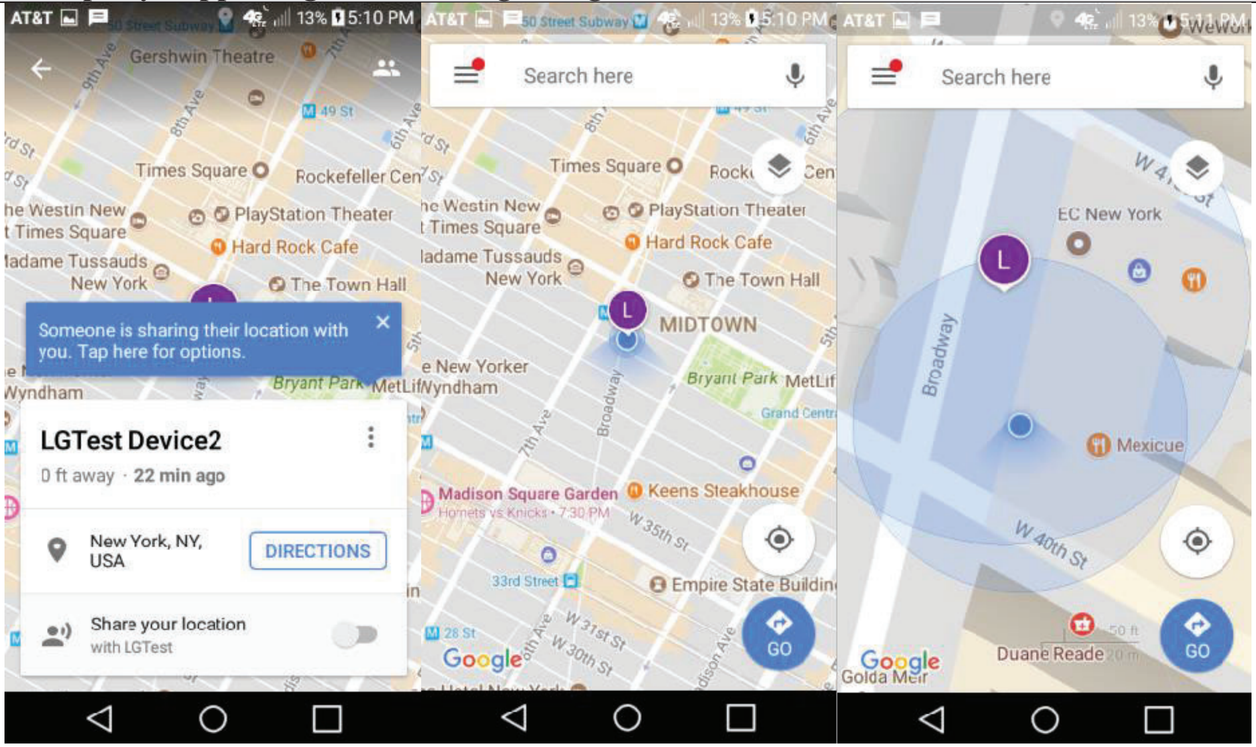
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 22% 12:01 PM www.google.com/ Find My Devic L Watson Ave Story Ave Lafayette Ave Brush Ave Target St Raym New LG Phoenix 3 Last seen just now 22% wallcricket PLAY SOUND **<u>Exemplary Google Maps Screenshots</u>**:

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

AT&T 13% 5:10 PM

Someone is sharing their location with you. Tap here for options.

LGTest Device2

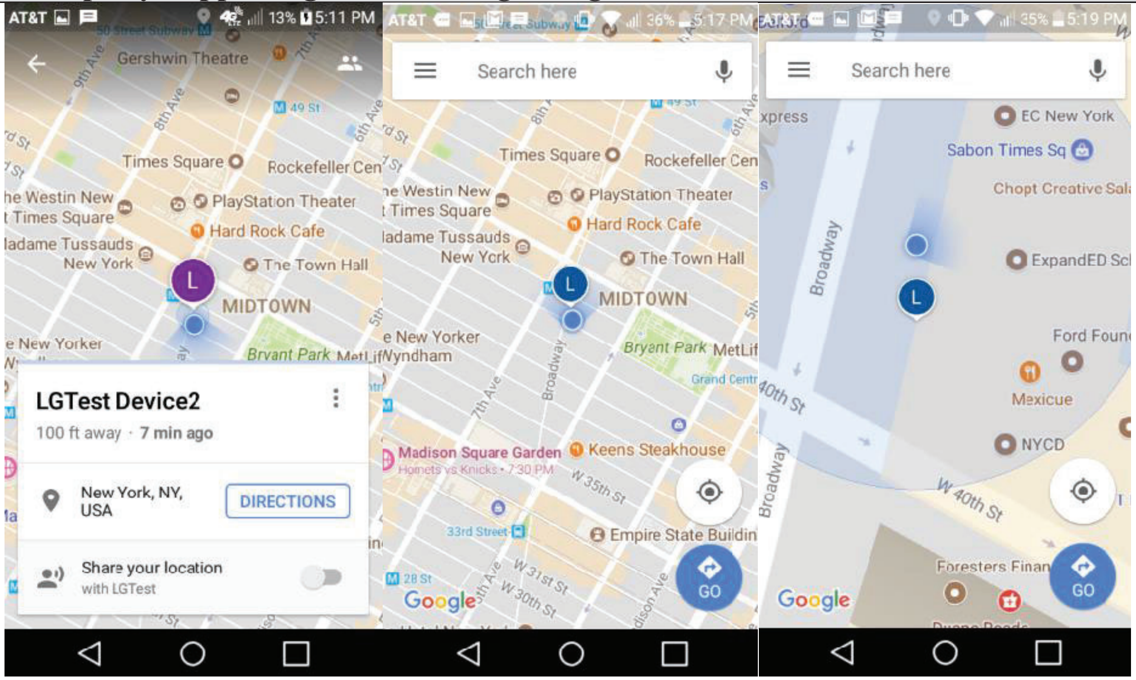
0 ft away · 22 min ago

New York, NY, USA

DIRECTIONS

Share your location with LGTest

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 LGTest Device2 100 ft away · 7 min ago New York, NY, USA DIRECTIONS Share your location with LGTest

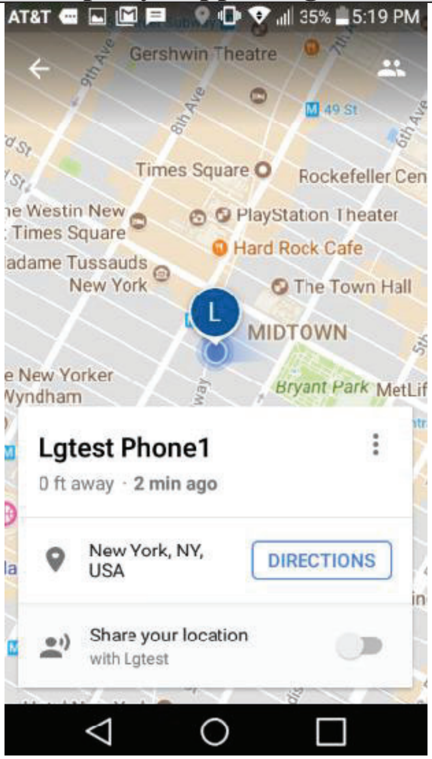
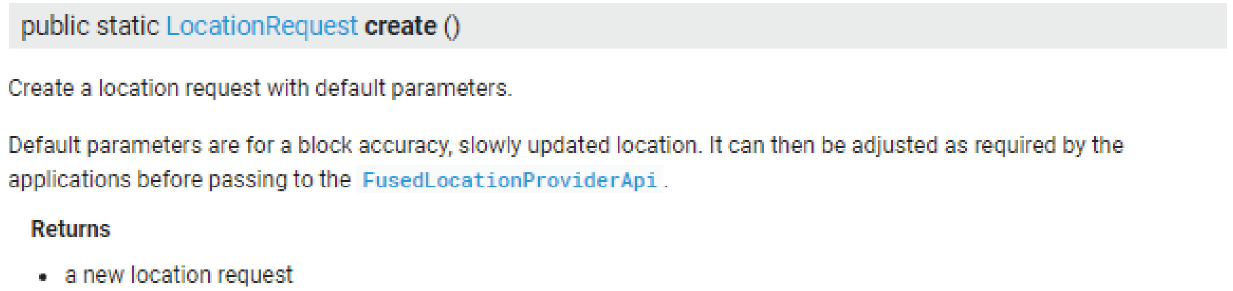
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 35% 5:19 PM Gershwin Theatre 9th Ave 8th Ave 7th Ave 6th Ave 49 St Times Square Rockefeller Cen The Westin New York at Times Square PlayStation Theater Hard Rock Cafe Madame Tussauds New York The Town Hall L MIDTOWN The New Yorker Wyndham Bryant Park MetLif Lgtest Phone1 0 ft away · 2 min ago New York, NY, USA DIRECTIONS Share your location with Lgtest **<u>Exemplary Source Code</u>**: The above functionality is performed at least in part by the following publicly available source code and/or source code that invokes or is invoked by the following source code (or a substantially similar copy compiled and loaded onto the Accused Products by ZTE). AGIS reserves the right to supplement these contentions with additional source code as discovery progresses and as additional source code is made available.

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>


public static LocationRequest **create** ()

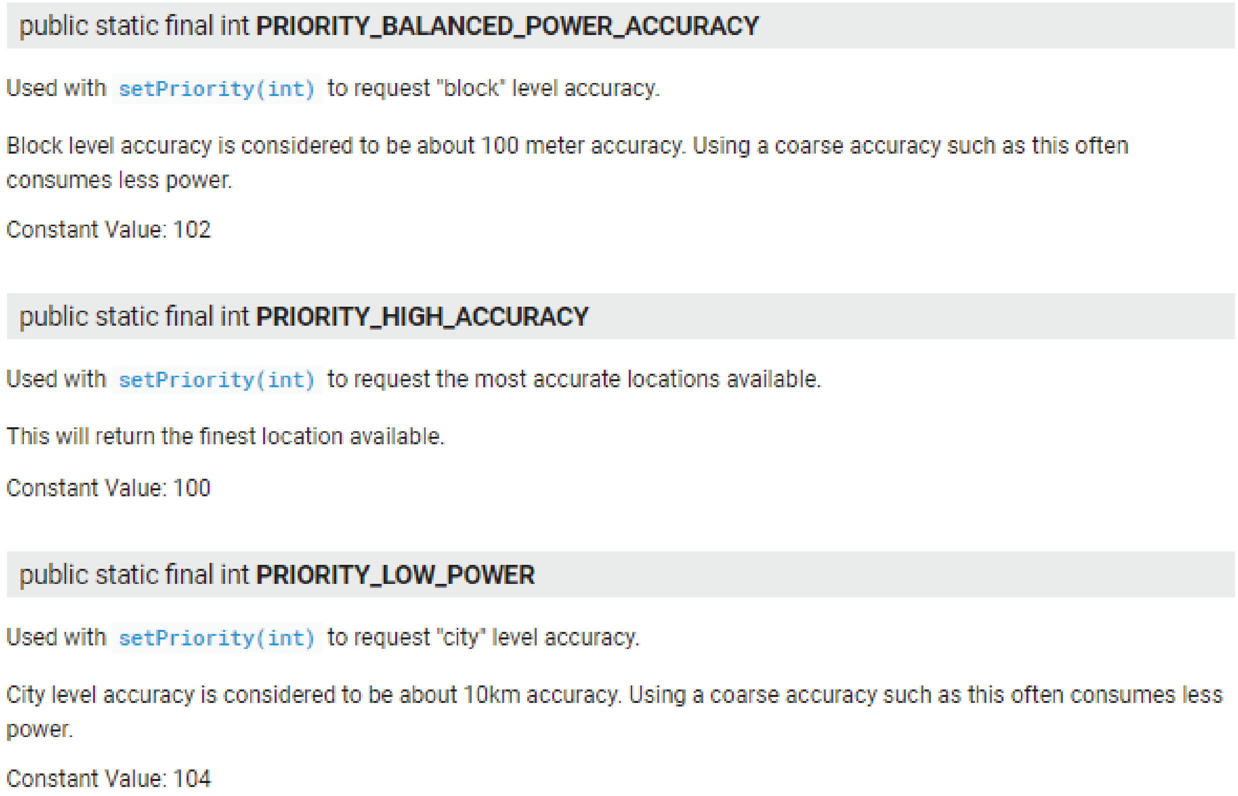
Create a location request with default parameters.

Default parameters are for a block accuracy, slowly updated location. It can then be adjusted as required by the applications before passing to the `FusedLocationProviderApi` .

Returns

- a new location request

https://developers.google.com/android/reference/com/google/android/gms/location/LocationRequest



public static final int **PRIORITY_BALANCED_POWER_ACCURACY**

Used with `setPriority(int)` to request "block" level accuracy.

Block level accuracy is considered to be about 100 meter accuracy. Using a coarse accuracy such as this often consumes less power.

Constant Value: 102

public static final int **PRIORITY_HIGH_ACCURACY**

Used with `setPriority(int)` to request the most accurate locations available.

This will return the finest location available.

Constant Value: 100

public static final int **PRIORITY_LOW_POWER**

Used with `setPriority(int)` to request "city" level accuracy.

City level accuracy is considered to be about 10km accuracy. Using a coarse accuracy such as this often consumes less power.

Constant Value: 104

https://developers.google.com/android/reference/com/google/android/gms/location/LocationRequest
</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

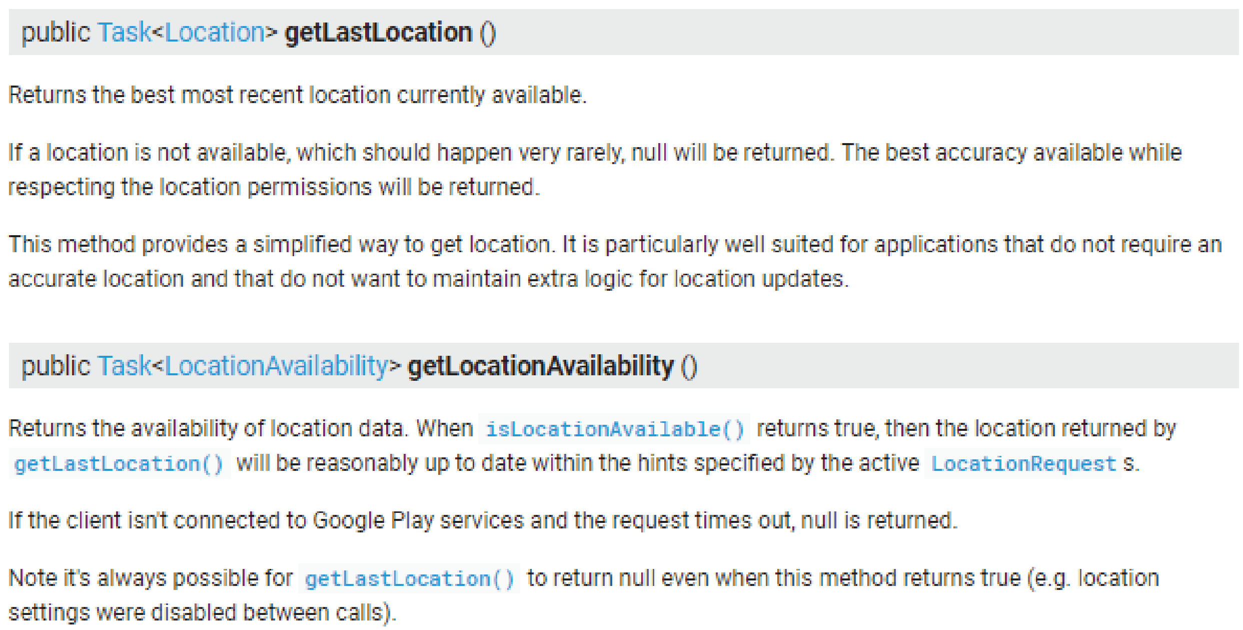
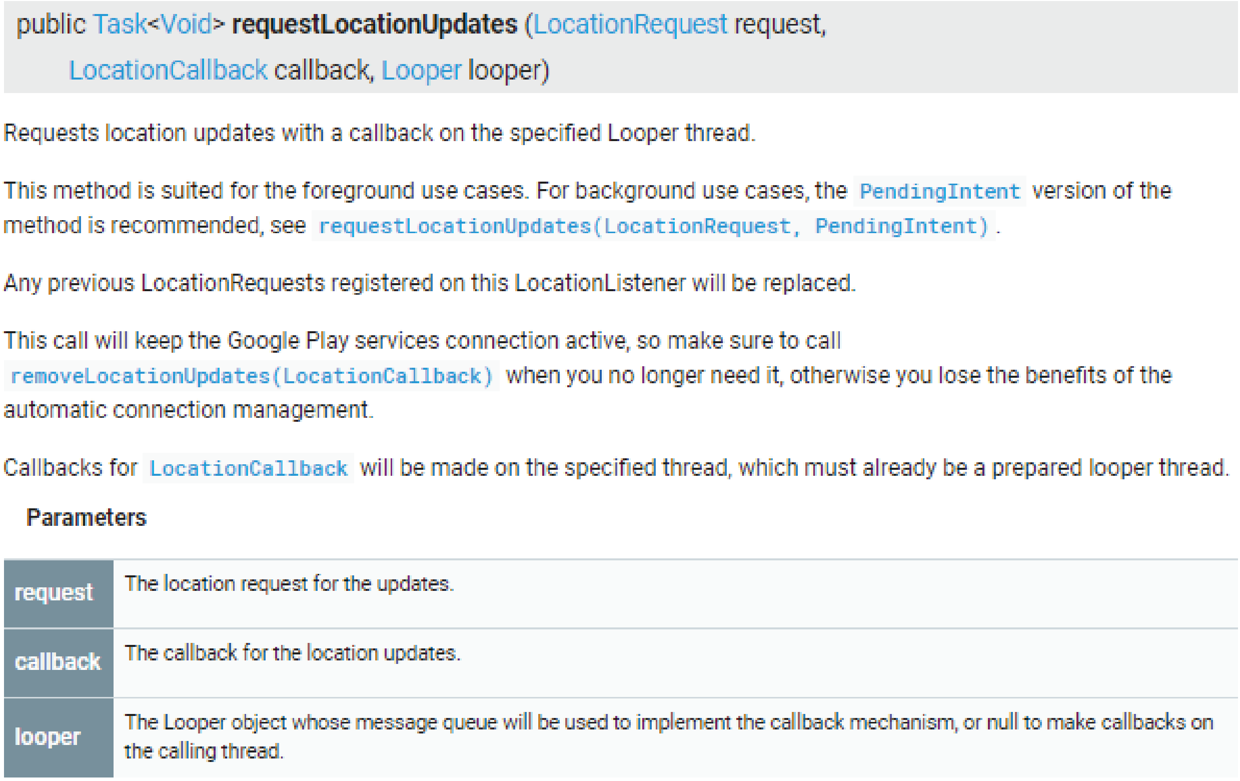
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public Task<Location> **getLastLocation** () Returns the best most recent location currently available. If a location is not available, which should happen very rarely, null will be returned. The best accuracy available while respecting the location permissions will be returned. This method provides a simplified way to get location. It is particularly well suited for applications that do not require an accurate location and that do not want to maintain extra logic for location updates. public Task<LocationAvailability> **getLocationAvailability** () Returns the availability of location data. When `isLocationAvailable()` returns true, then the location returned by `getLastLocation()` will be reasonably up to date within the hints specified by the active `LocationRequest` s. If the client isn't connected to Google Play services and the request times out, null is returned. Note it's always possible for `getLastLocation()` to return null even when this method returns true (e.g. location settings were disabled between calls). https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>


public Task<Void> **requestLocationUpdates** (LocationRequest request,
LocationCallback callback, Looper looper)

Requests location updates with a callback on the specified Looper thread.

This method is suited for the foreground use cases. For background use cases, the `PendingIntent` version of the method is recommended, see `requestLocationUpdates(LocationRequest, PendingIntent)`.

Any previous LocationRequests registered on this LocationListener will be replaced.

This call will keep the Google Play services connection active, so make sure to call `removeLocationUpdates(LocationCallback)` when you no longer need it, otherwise you lose the benefits of the automatic connection management.

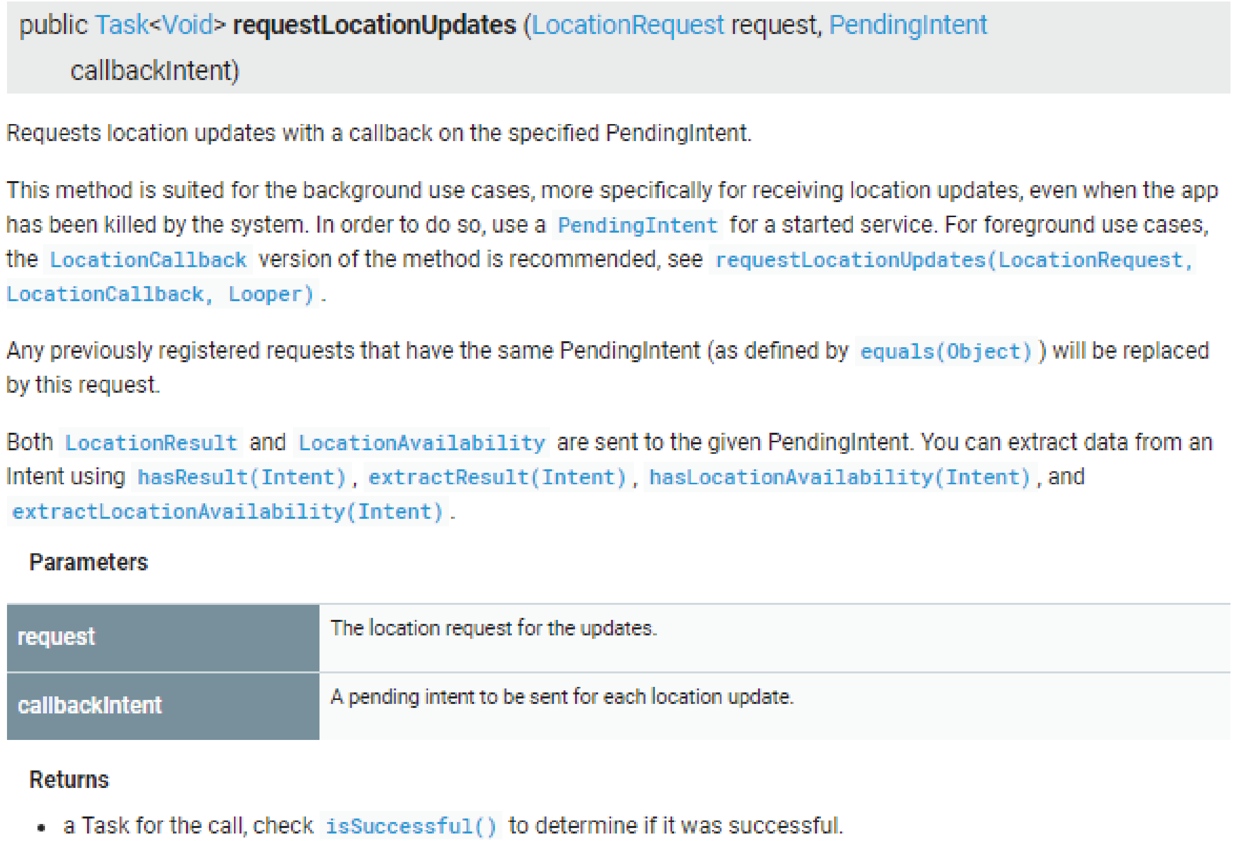
Callbacks for `LocationCallback` will be made on the specified thread, which must already be a prepared looper thread.

Parameters

<table>
<tr><td><b>request</b></td><td>The location request for the updates.</td></tr>
<tr><td><b>callback</b></td><td>The callback for the location updates.</td></tr>
<tr><td><b>looper</b></td><td>The Looper object whose message queue will be used to implement the callback mechanism, or null to make callbacks on the calling thread.</td></tr>
</table>

https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient
</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient

public Task<Void> **requestLocationUpdates** (LocationRequest request, PendingIntent callbackIntent)

Requests location updates with a callback on the specified PendingIntent.

This method is suited for the background use cases, more specifically for receiving location updates, even when the app has been killed by the system. In order to do so, use a `PendingIntent` for a started service. For foreground use cases, the `LocationCallback` version of the method is recommended, see `requestLocationUpdates(LocationRequest, LocationCallback, Looper)`.

Any previously registered requests that have the same PendingIntent (as defined by `equals(Object)`) will be replaced by this request.

Both `LocationResult` and `LocationAvailability` are sent to the given PendingIntent. You can extract data from an Intent using `hasResult(Intent)`, `extractResult(Intent)`, `hasLocationAvailability(Intent)`, and `extractLocationAvailability(Intent)`.

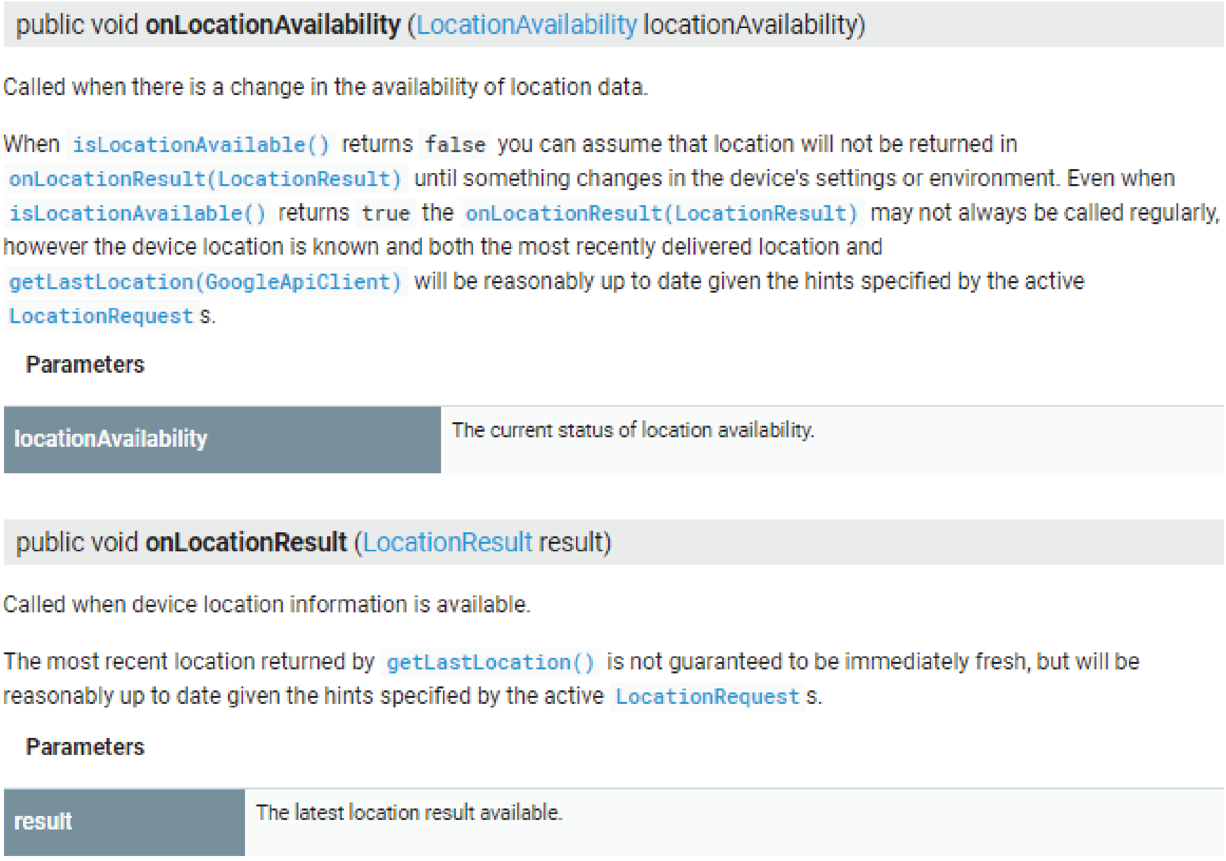
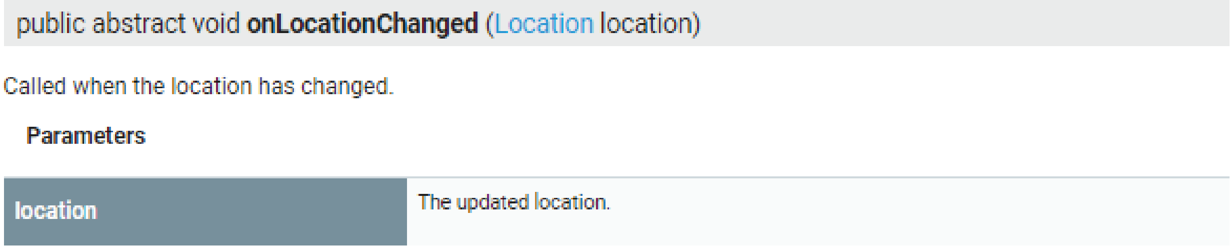
Parameters

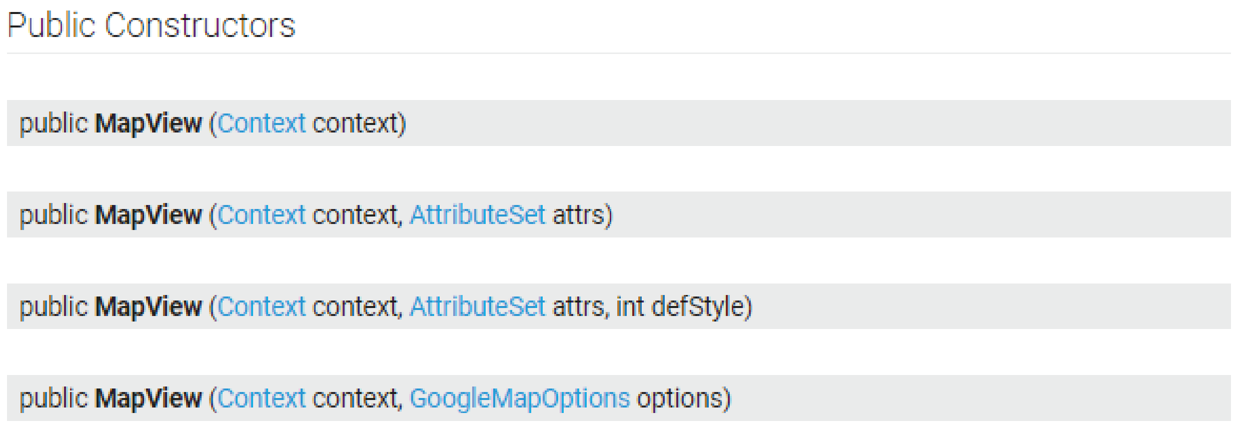
request	The location request for the updates.
callbackIntent	A pending intent to be sent for each location update.

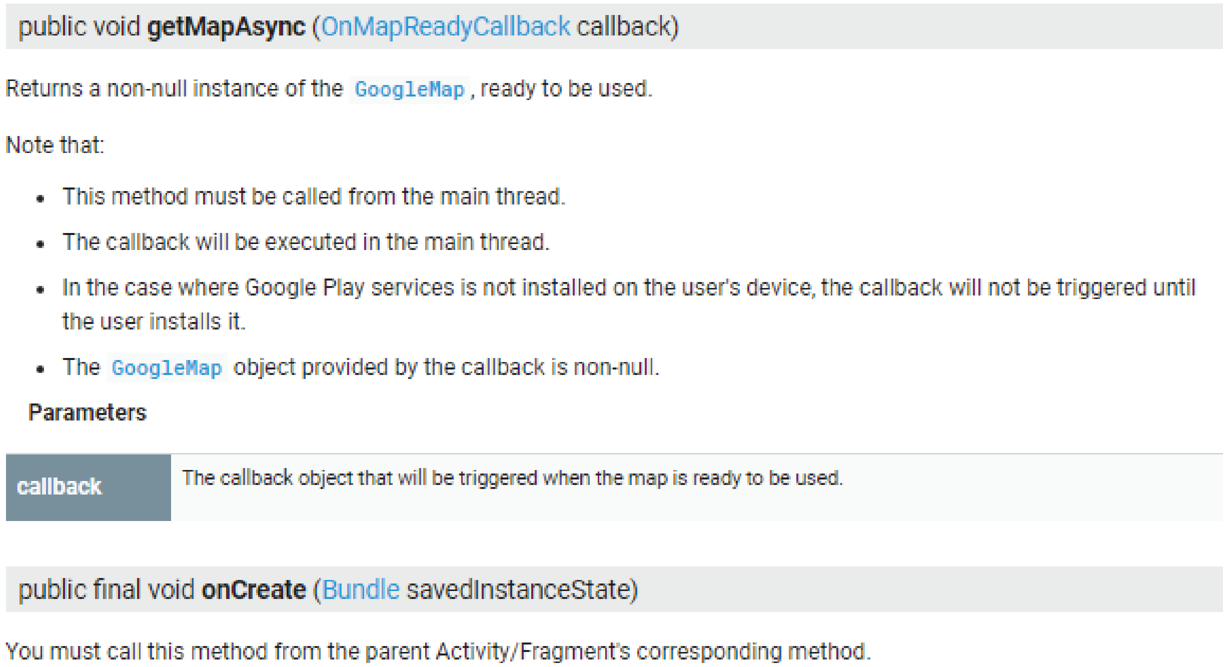
Returns

- a Task for the call, check `isSuccessful()` to determine if it was successful.

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public void **onLocationAvailability** (LocationAvailability locationAvailability) Called when there is a change in the availability of location data. When `isLocationAvailable()` returns `false` you can assume that location will not be returned in `onLocationResult(LocationResult)` until something changes in the device's settings or environment. Even when `isLocationAvailable()` returns `true` the `onLocationResult(LocationResult)` may not always be called regularly, however the device location is known and both the most recently delivered location and `getLastLocation(GoogleApiClient)` will be reasonably up to date given the hints specified by the active `LocationRequest` s. **Parameters** **locationAvailability** The current status of location availability. public void **onLocationResult** (LocationResult result) Called when device location information is available. The most recent location returned by `getLastLocation()` is not guaranteed to be immediately fresh, but will be reasonably up to date given the hints specified by the active `LocationRequest` s. **Parameters** **result** The latest location result available. https://developers.google.com/android/reference/com/google/android/gms/location/LocationCallback  public abstract void **onLocationChanged** (Location location) Called when the location has changed. **Parameters** **location** The updated location.

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	https://developers.google.com/android/reference/com/google/android/gms/location/LocationListener  Public Constructors public **MapView** (Context context) public **MapView** (Context context, AttributeSet attrs) public **MapView** (Context context, AttributeSet attrs, int defStyle) public **MapView** (Context context, GoogleMapOptions options) https://developers.google.com/android/reference/com/google/android/gms/maps/MapView

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public void **getMapAsync** (OnMapReadyCallback callback) Returns a non-null instance of the `GoogleMap`, ready to be used. Note that: • This method must be called from the main thread. • The callback will be executed in the main thread. • In the case where Google Play services is not installed on the user's device, the callback will not be triggered until the user installs it. • The `GoogleMap` object provided by the callback is non-null. **Parameters** callback — The callback object that will be triggered when the map is ready to be used. public final void **onCreate** (Bundle savedInstanceState) You must call this method from the parent Activity/Fragment's corresponding method. https://developers.google.com/android/reference/com/google/android/gms/maps/MapView
[1E] sending, to a second server, a request for second georeferenced map data different from the first georeferenced map data;	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of: sending, to a second server, a request for second georeferenced map data different from the first georeferenced map data. **Regarding Find My Device** and Android Device Manager, the Accused Products display, to the user on the display of the first device, a map with one or more symbols corresponding to one or more second devices. The user, via the first device, or the device itself requests different map data from a second server. The request occurs responsive to user input (e.g., zoom, drag, pan, change focus, change map type, symbol or device selection, another device or user selection, refresh or reload request, change in position of first device, change in position of a second device). Alternatively, the request occurs responsive to an

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	automatic and/or pre-determined control caused by an instruction from within the first device or from the one or more second devices, e.g. a refresh. The different map data includes an update to the first data or a replacement of the first data. Upon information and belief, the new map data may come from one of many sources and servers. **<u>Regarding Google Maps</u>**, Google Latitude, Google Plus, Google Hangouts, Google Messages, Google Allo, Google Duo, Google Chrome, and Android Messenger, the Accused Products display, to the user on the display of the first device, a map with one or more symbols corresponding to one or more second users (or second devices corresponding to the second users). The user, via the first device, or the device itself requests different map data from a second server. The request occurs responsive to user input (e.g., zoom, drag, pan, change focus, change map type, refresh or reload request, symbol or device selection, another device or user selection, change in position of first device, change in position of a second device). Alternatively, the request occurs responsive to an automatic and/or pre-determined control caused by an instruction from within the first device or from the one or more second devices, e.g. a refresh. The different map data includes an update to the first data or a replacement of the first data. Upon information and belief, the new map data may come from one of many sources and servers. **<u>Exemplary Support for Find My Device</u>**:

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

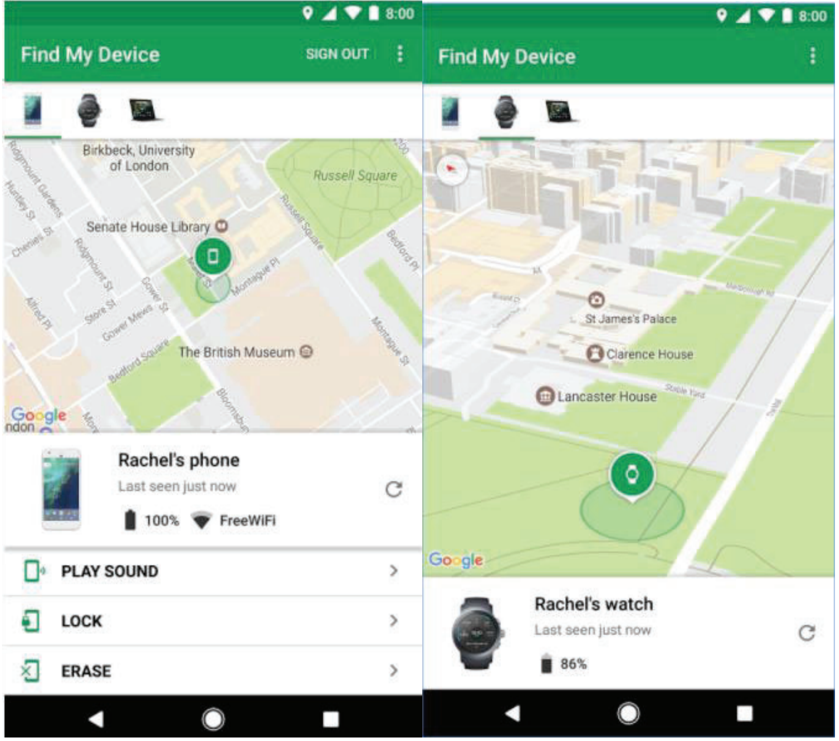
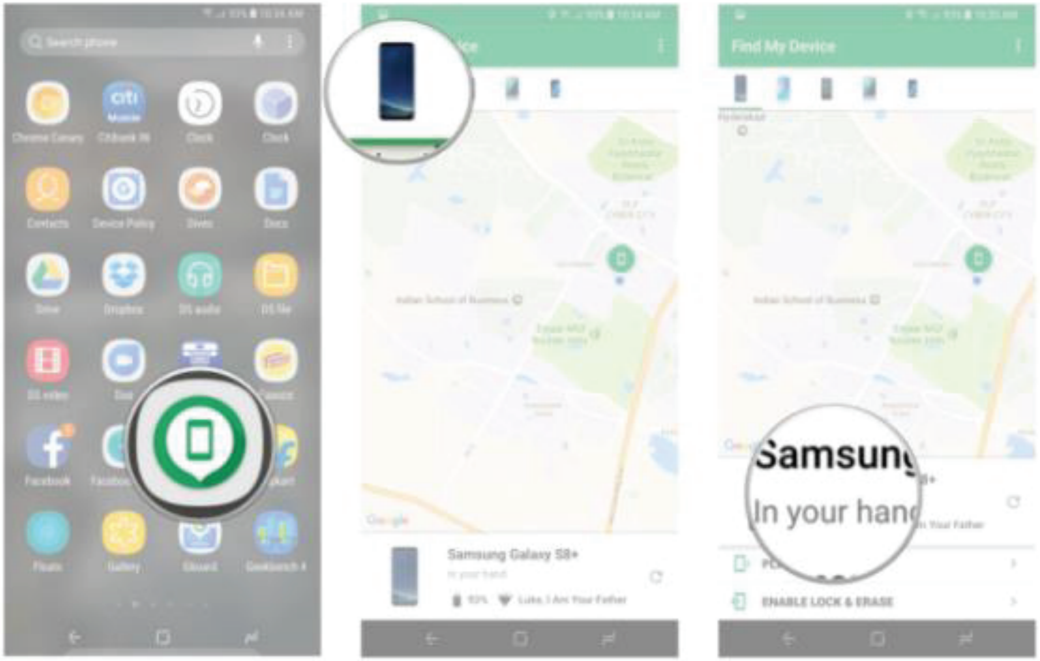
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 8:00 Find My Device SIGN OUT Birkbeck, University of London Russell Square Senate House Library The British Museum Google Rachel's phone Last seen just now 100% FreeWiFi PLAY SOUND LOCK ERASE 8:00 Find My Device St James's Palace Clarence House Lancaster House Google Rachel's watch Last seen just now 86% https://play.google.com/store/apps/details?id=com.google.android.apps.adm&hl=en

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	Once you're signed in to Find My Device, you'll see a map with your current location as well as the make and model of your phone, and two options — Play Sound, and Enable Lock & Erase. Hitting the latter option will allow you to start using the Lock and Erase functions. If you've signed into more than one phone, you can select a particular device by browsing the list at the top of the screen. 1. Open **Find My Device** from your home screen or app drawer. 2. Select your phone from the **list of devices at the top of the screen**. 3. See if your phone is **discoverable**.  Find My Device Samsung Galaxy S8+ Samsung In your hand ENABLE LOCK & ERASE https://www.androidcentral.com/find-my-device

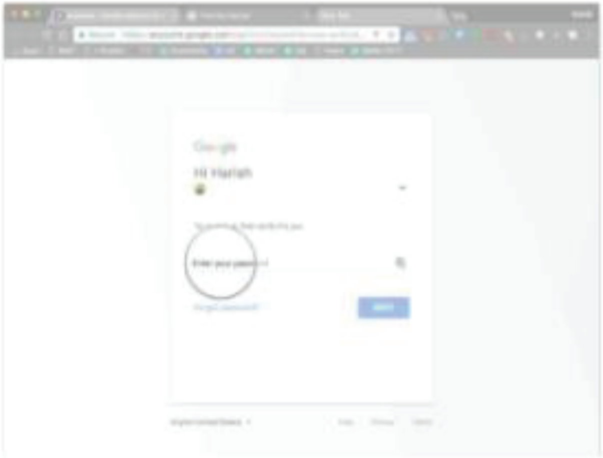
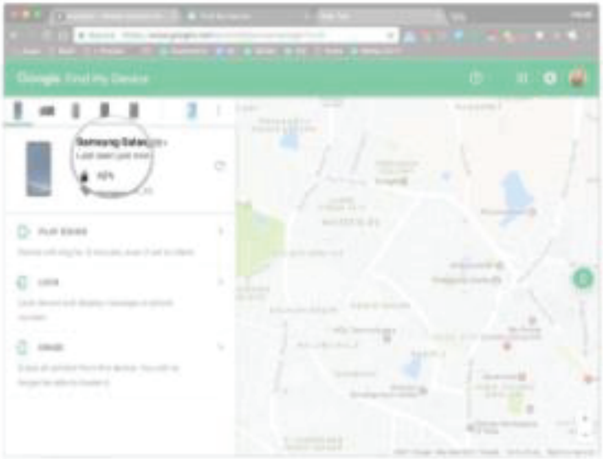
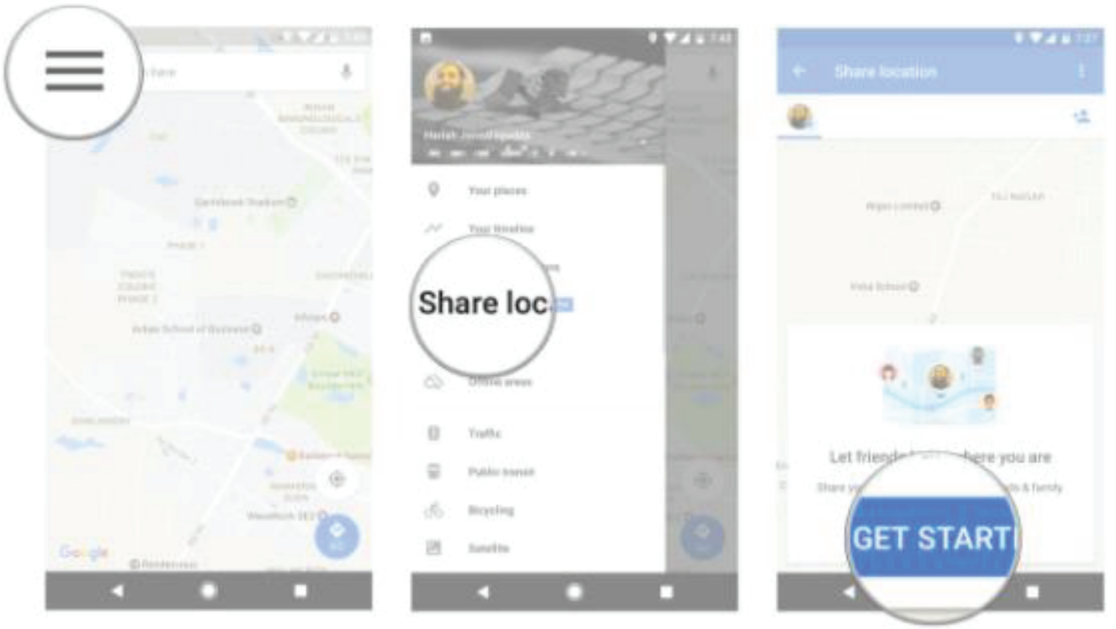
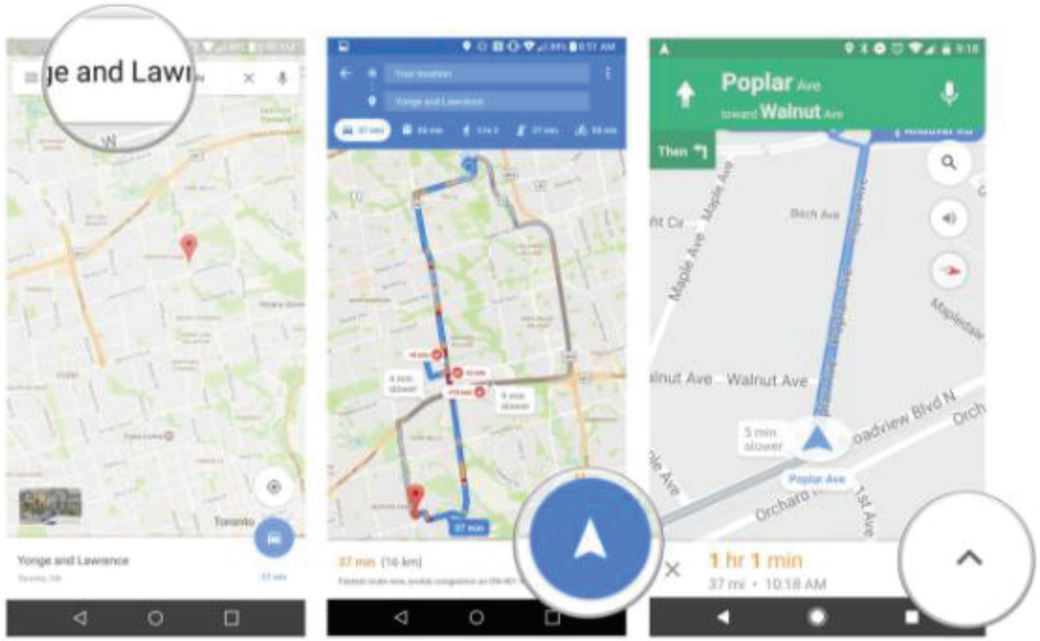
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to locate your phone over the internet If you've lost your phone, you can remotely locate it through the Find My Device website. You'll need to sign in to the Google account that was used to set up Find My Device. It takes a few seconds, but the service should be able to track your phone. Alternatively, you can also do a Google search for "find my phone" to locate your handset. 1. Head to the **Find My Device website**. 2. Sign in to your **Google account**. 3. Check if your device is **visible**.   https://www.androidcentral.com/find-my-device **Exemplary Support for Google Maps**:

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to create a shareable link You can also create a link and use it to share your location easily. Here's how to do it: 1. Tap the **hamburger menu** on the top left corner of the screen. 2. Select **Share location**. 3. Tap **Get Started**.  Share loc Your places Your timeline Offline areas Traffic Public transit Bicycling Satellite Share location Let friends ... where you are GET START https://www.androidcentral.com/how-share-location-google-maps

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to share your navigation directions while you walk, drive or transit One of the best ways to share your location with someone is during a drive, walk or transit. If you're meeting a friend or family member somewhere, or navigating towards their home, Google Maps lets you share your location with them for the duration of the trip. It's magic! 1. In the **search bar** enter your destination. 2. Pick your navigation type (drive, transit, walk) and press the blue **navigate button**. 3. Tap the arrow next to the **time-to-destination** number at the bottom of the screen.  je and Lawr Yonge and Lawrence Toronto 37 min (16 km) 37 min Poplar Ave toward Walnut Ave Then Maple Ave Birch Ave Walnut Ave 5 min slower Poplar Ave Orchard 1st Ave 1 hr 1 min 37 mi · 10:18 AM https://www.androidcentral.com/how-share-location-google-maps

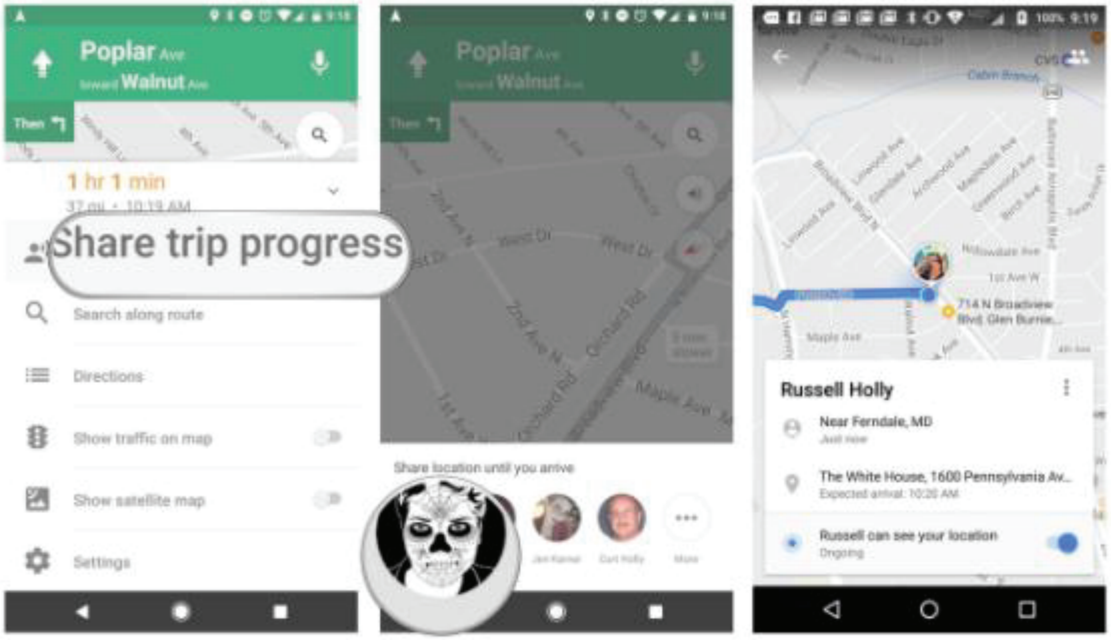
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	4. Tap Share trip progress. 5. Choose one or more contacts to share trip progress.  Poplar Ave toward Walnut Ave Then 1 hr 1 min 37 mi · 10:19 AM Share trip progress Search along route Directions Show traffic on map Show satellite map Settings Share location until you arrive More Russell Holly Near Ferndale, MD Just now The White House, 1600 Pennsylvania Av... Expected arrival: 10:20 AM Russell can see your location Ongoing 714 N Broadview Blvd, Glen Burnie You can also stop sharing your location with someone before a trip ends. https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

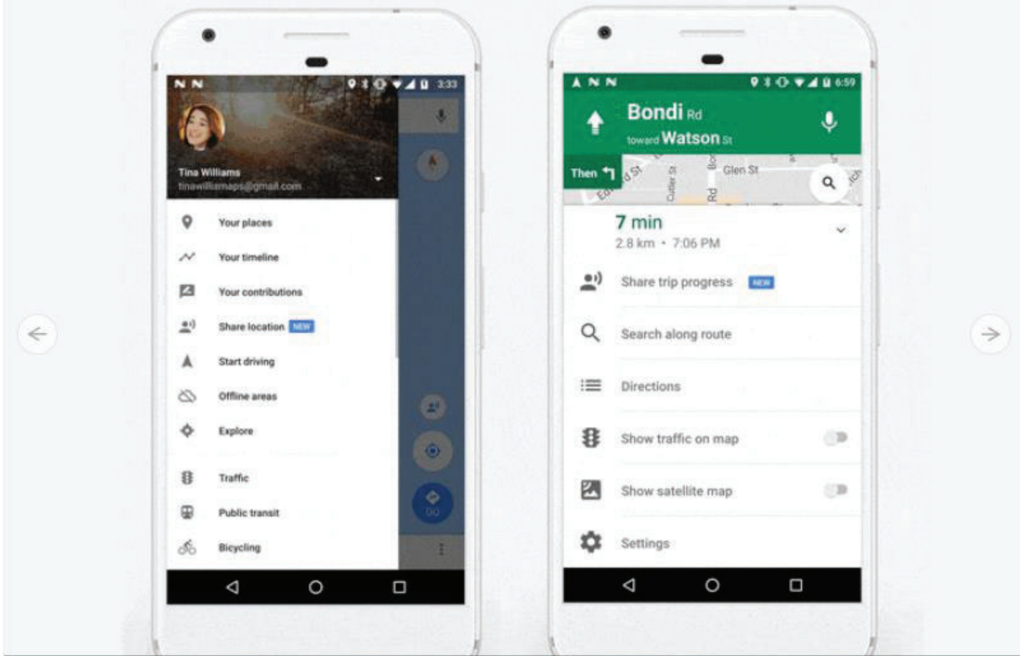
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Tina Williams tinawilliamsmaps@gmail.com Your places Your timeline Your contributions Share location NEW Start driving Offline areas Explore Traffic Public transit Bicycling Bondi Rd toward Watson St Then Glen St 7 min 2.8 km • 7:06 PM Share trip progress NEW Search along route Directions Show traffic on map Show satellite map Settings Ways to start location sharing: (left) from the app navigation drawer, and (right) from the menu, you can drag up while navigating somewhere. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

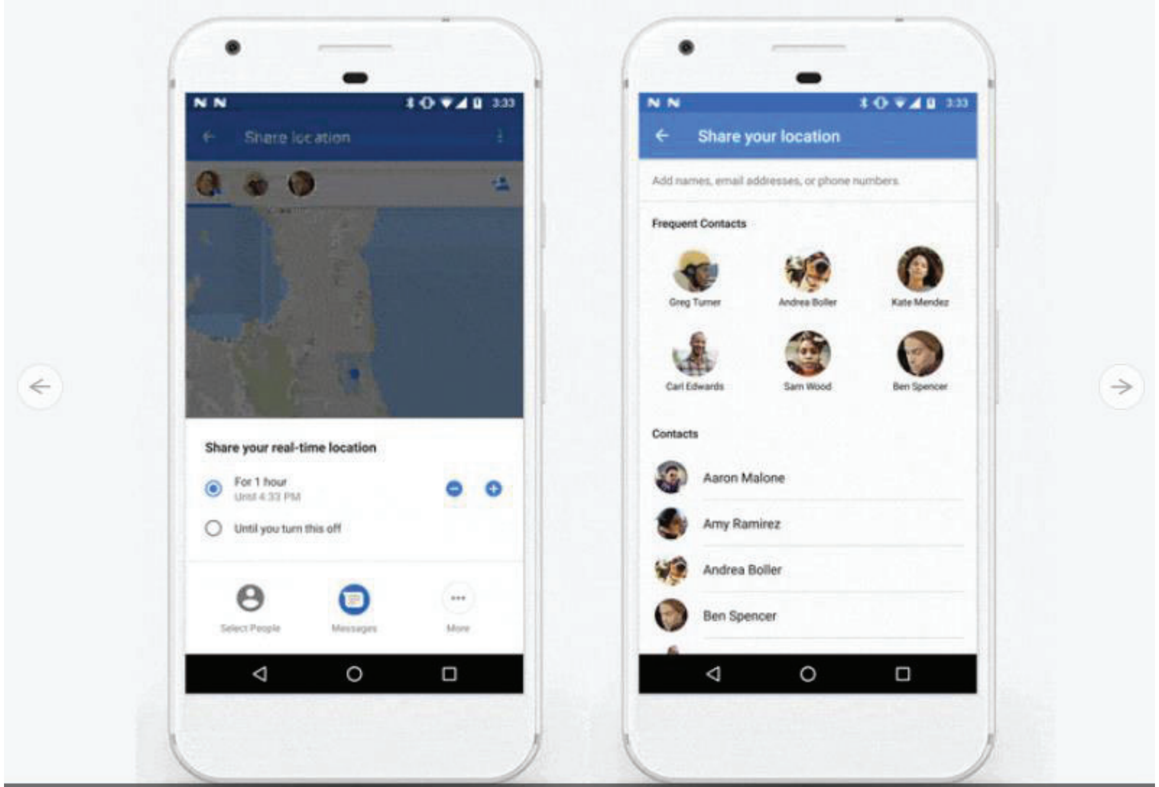
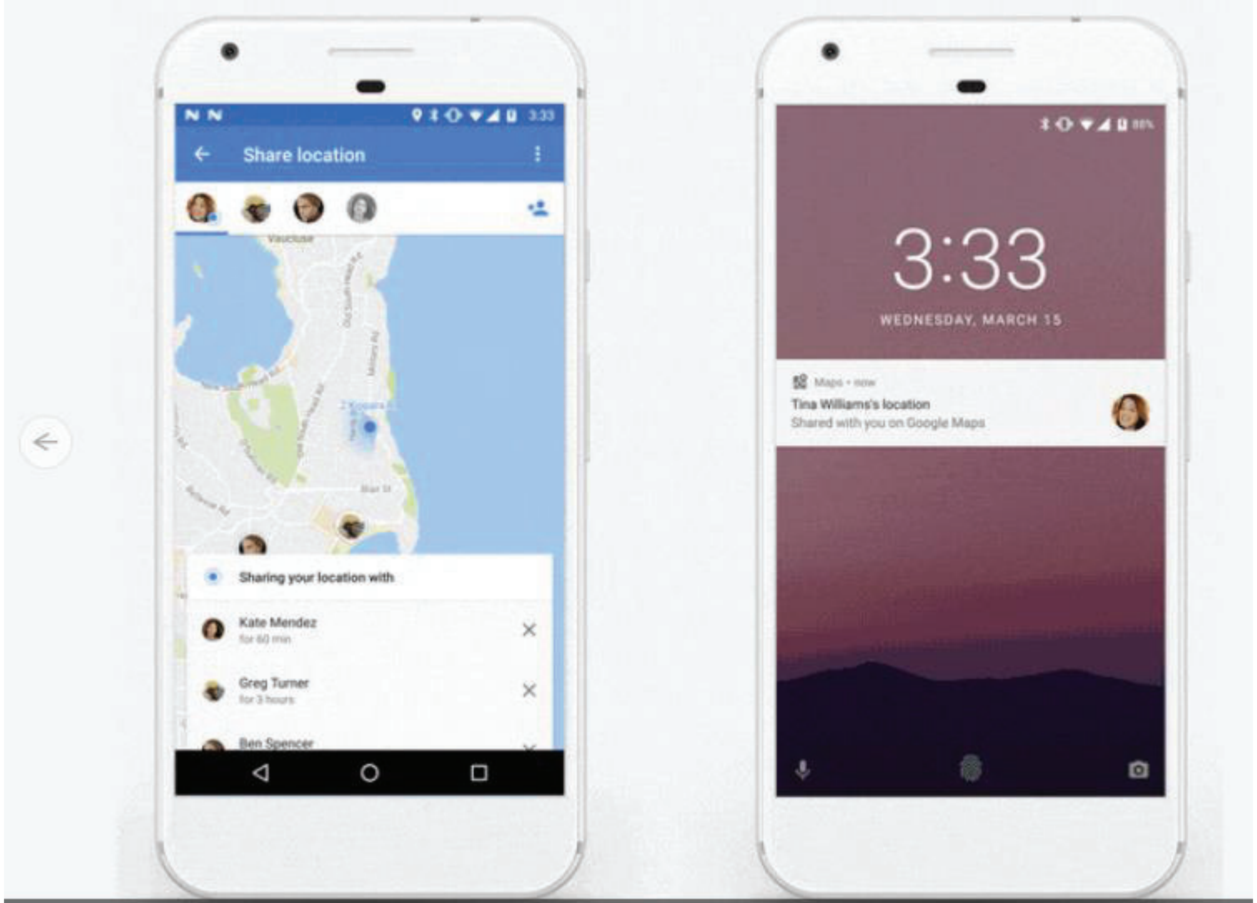
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Share location Share your real-time location For 1 hour Until 4:33 PM Until you turn this off Select People Messages More Share your location Add names, email addresses, or phone numbers Frequent Contacts Greg Turner Andrea Boller Kate Mendez Carl Edwards Sam Wood Ben Spencer Contacts Aaron Malone Amy Ramirez Andrea Boller Ben Spencer You can pick how long you want to share your location for, and with whom you want to share it. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Share location Sharing your location with Kate Mendez for 60 min Greg Turner for 3 hours Ben Spencer 3:33 WEDNESDAY, MARCH 15 Maps • now Tina Williams's location Shared with you on Google Maps People you've shared with will get a notification from Google Maps. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

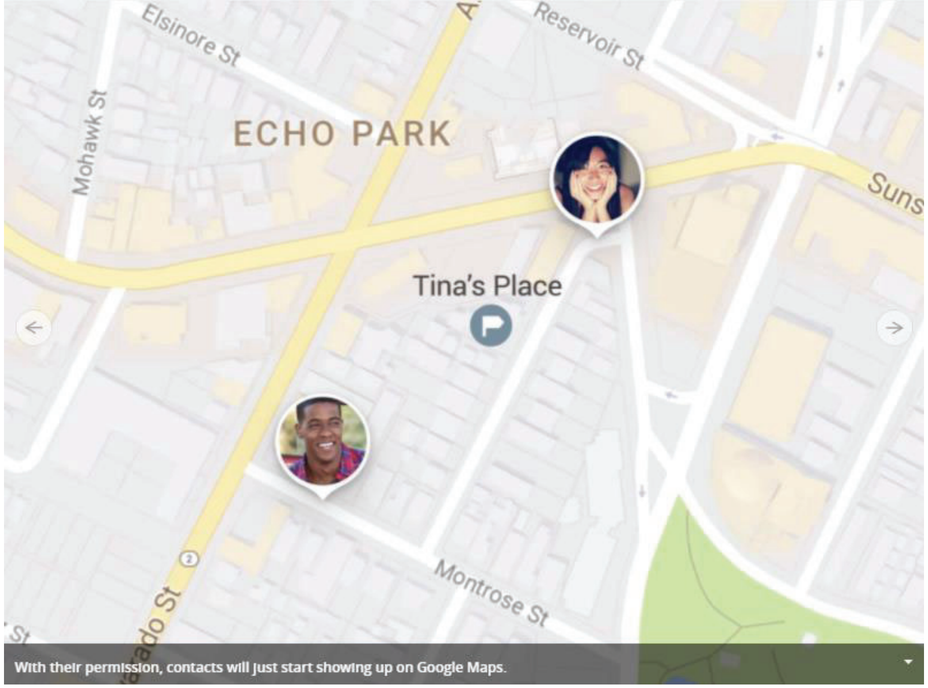
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Elsinore St Reservoir St Mohawk St ECHO PARK Suns Tina's Place Montrose St ado St St With their permission, contacts will just start showing up on Google Maps. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/ For example, Google requests and receives map data from many sources and servers, both internally and externally. In response to one of a number of user actions (e.g. zoom, drag or change focus, change map type, select another device or user), new map data is retrieved to complete the user's action and the displayed map is replaced or updated accordingly. The new map data may come from one of many sources and servers. The Accused Products request and receive the map data described above. Alternatively, any computer signed-in to the Google network services may request and receive the map data described above. Another example of this limitation in practice includes toggling between map types. The first-displayed map can be the standard interface map with geographical points of interest. In response to user action, a second map can be

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

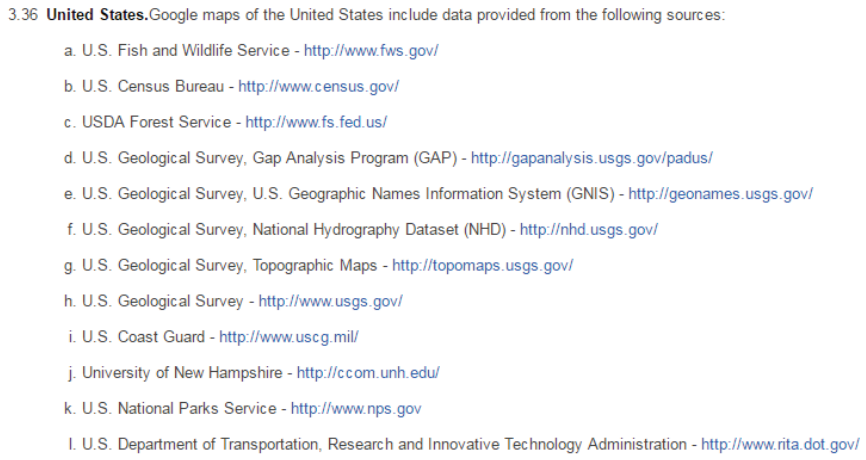
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	called with satellite imagery, which would be retrieved from a different server source having satellite information. Another known example is when a user zooms into a standard map and a transmit map is retrieved and overlaid or otherwise included to form a second map with transit data pulled from a transit data server.  3.36 **United States.**Google maps of the United States include data provided from the following sources: a. U.S. Fish and Wildlife Service - http://www.fws.gov/ b. U.S. Census Bureau - http://www.census.gov/ c. USDA Forest Service - http://www.fs.fed.us/ d. U.S. Geological Survey, Gap Analysis Program (GAP) - http://gapanalysis.usgs.gov/padus/ e. U.S. Geological Survey, U.S. Geographic Names Information System (GNIS) - http://geonames.usgs.gov/ f. U.S. Geological Survey, National Hydrography Dataset (NHD) - http://nhd.usgs.gov/ g. U.S. Geological Survey, Topographic Maps - http://topomaps.usgs.gov/ h. U.S. Geological Survey - http://www.usgs.gov/ i. U.S. Coast Guard - http://www.uscg.mil/ j. University of New Hampshire - http://ccom.unh.edu/ k. U.S. National Parks Service - http://www.nps.gov l. U.S. Department of Transportation, Research and Innovative Technology Administration - http://www.rita.dot.gov/ https://www.google.com/intl/en_us/help/legalnotices_maps.html

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

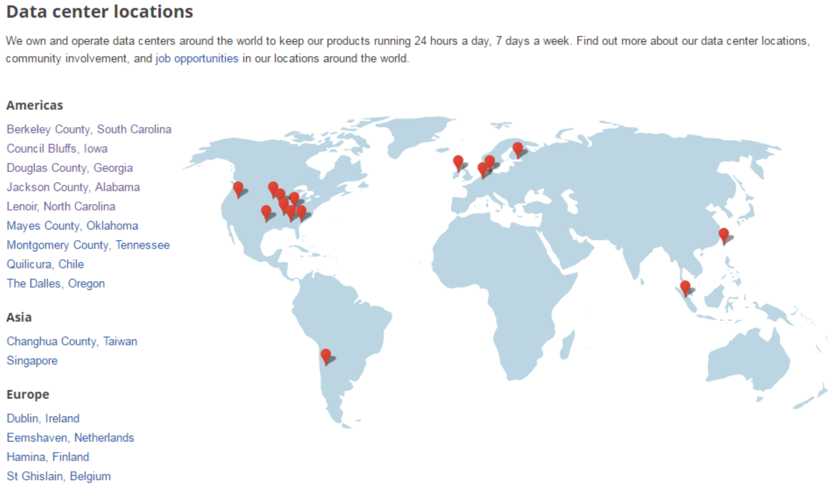
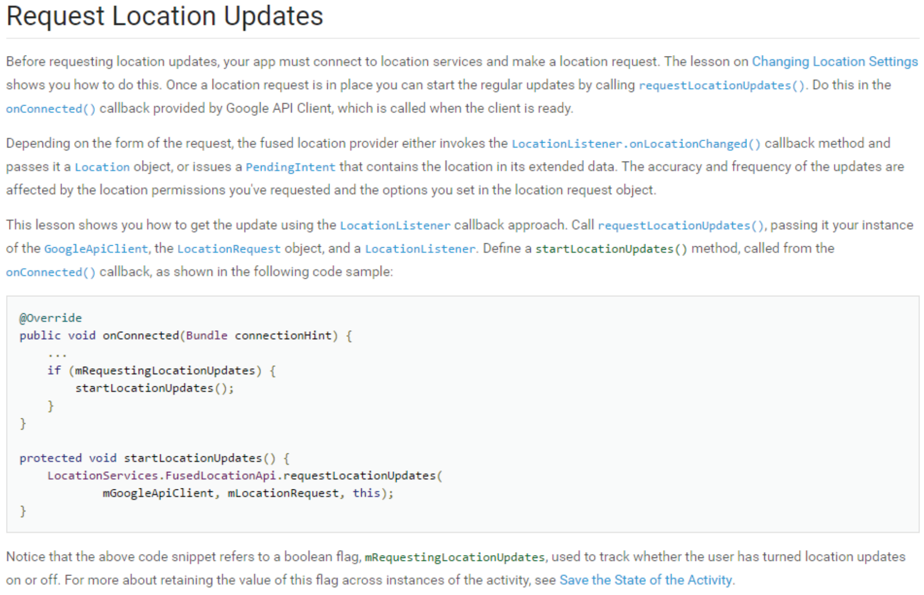
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 **Data center locations** We own and operate data centers around the world to keep our products running 24 hours a day, 7 days a week. Find out more about our data center locations, community involvement, and job opportunities in our locations around the world. **Americas** Berkeley County, South Carolina Council Bluffs, Iowa Douglas County, Georgia Jackson County, Alabama Lenoir, North Carolina Mayes County, Oklahoma Montgomery County, Tennessee Quilicura, Chile The Dalles, Oregon **Asia** Changhua County, Taiwan Singapore **Europe** Dublin, Ireland Eemshaven, Netherlands Hamina, Finland St Ghislain, Belgium https://www.google.com/about/datacenters/inside/locations/index.html

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

Request Location Updates

Before requesting location updates, your app must connect to location services and make a location request. The lesson on Changing Location Settings shows you how to do this. Once a location request is in place you can start the regular updates by calling `requestLocationUpdates()`. Do this in the `onConnected()` callback provided by Google API Client, which is called when the client is ready.

Depending on the form of the request, the fused location provider either invokes the `LocationListener.onLocationChanged()` callback method and passes it a `Location` object, or issues a `PendingIntent` that contains the location in its extended data. The accuracy and frequency of the updates are affected by the location permissions you've requested and the options you set in the location request object.

This lesson shows you how to get the update using the `LocationListener` callback approach. Call `requestLocationUpdates()`, passing it your instance of the `GoogleApiClient`, the `LocationRequest` object, and a `LocationListener`. Define a `startLocationUpdates()` method, called from the `onConnected()` callback, as shown in the following code sample:

```
@Override
public void onConnected(Bundle connectionHint) {
    ...
    if (mRequestingLocationUpdates) {
        startLocationUpdates();
    }
}

protected void startLocationUpdates() {
    LocationServices.FusedLocationApi.requestLocationUpdates(
            mGoogleApiClient, mLocationRequest, this);
}
```

Notice that the above code snippet refers to a boolean flag, `mRequestingLocationUpdates`, used to track whether the user has turned location updates on or off. For more about retaining the value of this flag across instances of the activity, see Save the State of the Activity.

https://developer.android.com/training/location/receive-location-updates.html

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

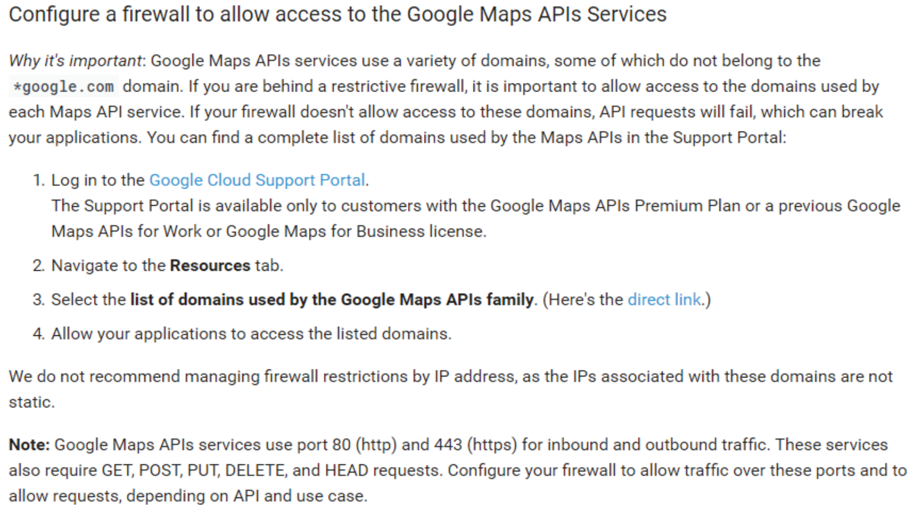
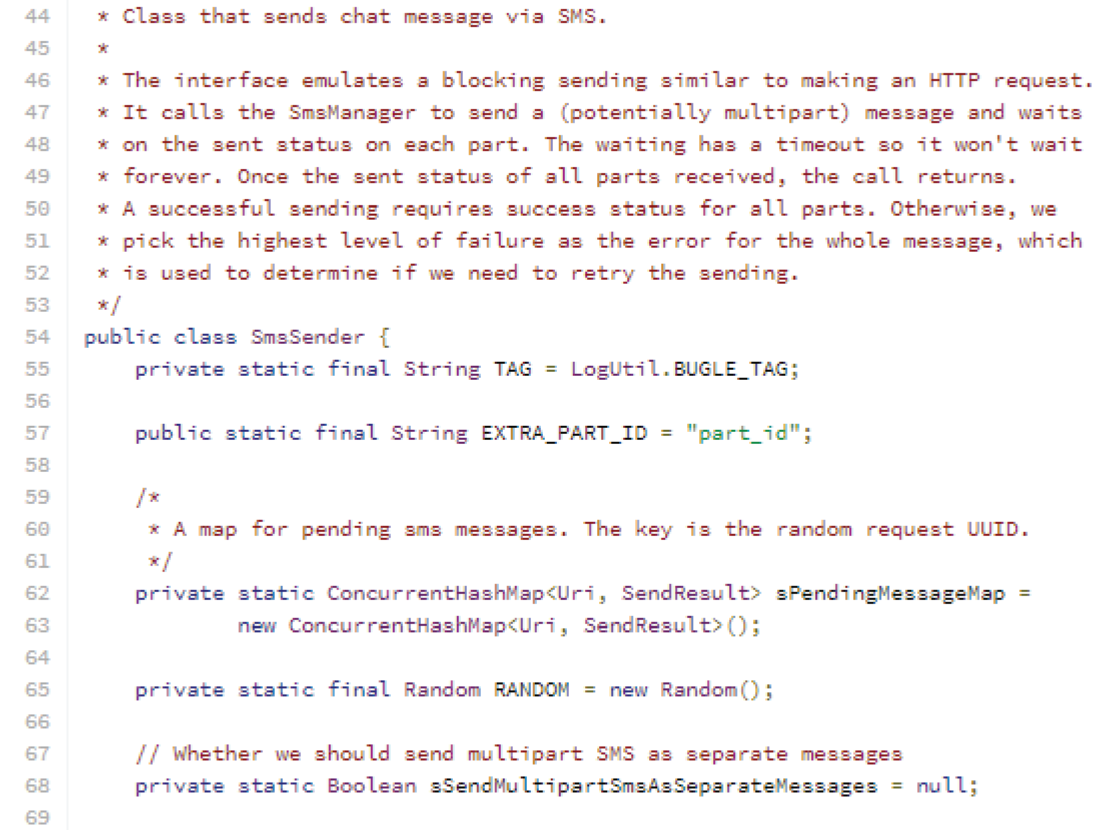
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Configure a firewall to allow access to the Google Maps APIs Services *Why it's important*: Google Maps APIs services use a variety of domains, some of which do not belong to the `*google.com` domain. If you are behind a restrictive firewall, it is important to allow access to the domains used by each Maps API service. If your firewall doesn't allow access to these domains, API requests will fail, which can break your applications. You can find a complete list of domains used by the Maps APIs in the Support Portal: 1. Log in to the Google Cloud Support Portal. The Support Portal is available only to customers with the Google Maps APIs Premium Plan or a previous Google Maps APIs for Work or Google Maps for Business license. 2. Navigate to the **Resources** tab. 3. Select the **list of domains used by the Google Maps APIs family**. (Here's the direct link.) 4. Allow your applications to access the listed domains. We do not recommend managing firewall restrictions by IP address, as the IPs associated with these domains are not static. **Note:** Google Maps APIs services use port 80 (http) and 443 (https) for inbound and outbound traffic. These services also require GET, POST, PUT, DELETE, and HEAD requests. Configure your firewall to allow traffic over these ports and to allow requests, depending on API and use case. https://developers.google.com/maps/premium/prelaunch-checklist#firewall **Exemplary Screenshots**: See, e.g., 1D and 1E above. **Exemplary Source Code**: The above functionality is performed at least in part by the following publicly available source code and/or source code that invokes or is invoked by the following source code (or a substantially similar copy compiled and loaded onto the Accused Products by ZTE). AGIS reserves the right to supplement these contentions with additional source code as discovery progresses and as additional source code is made available.

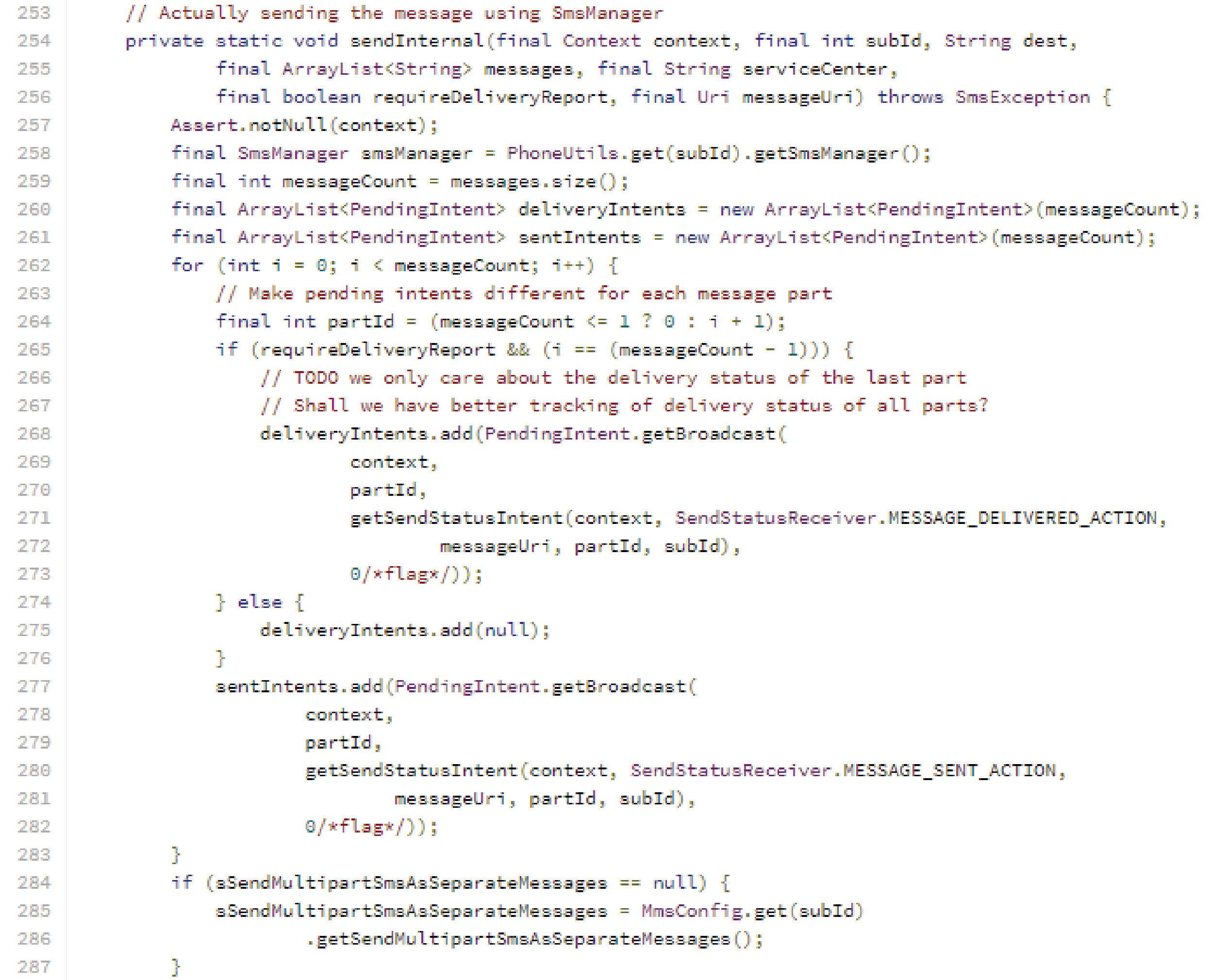
Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
 * Class that sends chat message via SMS.
 *
 * The interface emulates a blocking sending similar to making an HTTP request.
 * It calls the SmsManager to send a (potentially multipart) message and waits
 * on the sent status on each part. The waiting has a timeout so it won't wait
 * forever. Once the sent status of all parts received, the call returns.
 * A successful sending requires success status for all parts. Otherwise, we
 * pick the highest level of failure as the error for the whole message, which
 * is used to determine if we need to retry the sending.
 */
public class SmsSender {
    private static final String TAG = LogUtil.BUGLE_TAG;

    public static final String EXTRA_PART_ID = "part_id";

    /*
     * A map for pending sms messages. The key is the random request UUID.
     */
    private static ConcurrentHashMap<Uri, SendResult> sPendingMessageMap =
            new ConcurrentHashMap<Uri, SendResult>();

    private static final Random RANDOM = new Random();

    // Whether we should send multipart SMS as separate messages
    private static Boolean sSendMultipartSmsAsSeparateMessages = null;

```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/SmsSender.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

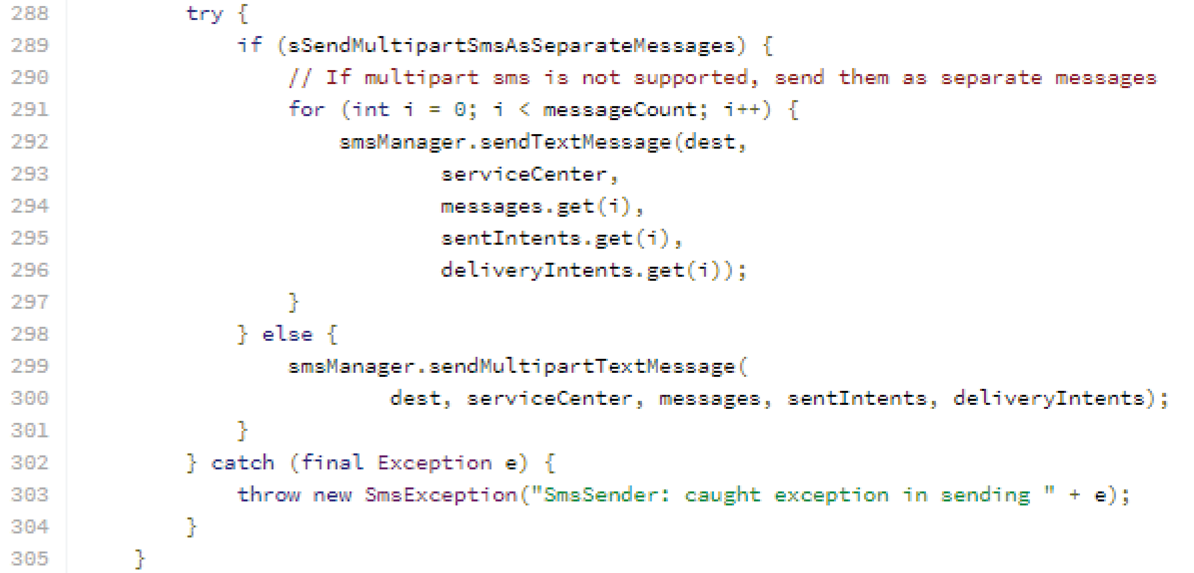
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
    // Actually sending the message using SmsManager
    private static void sendInternal(final Context context, final int subId, String dest,
            final ArrayList<String> messages, final String serviceCenter,
            final boolean requireDeliveryReport, final Uri messageUri) throws SmsException {
        Assert.notNull(context);
        final SmsManager smsManager = PhoneUtils.get(subId).getSmsManager();
        final int messageCount = messages.size();
        final ArrayList<PendingIntent> deliveryIntents = new ArrayList<PendingIntent>(messageCount);
        final ArrayList<PendingIntent> sentIntents = new ArrayList<PendingIntent>(messageCount);
        for (int i = 0; i < messageCount; i++) {
            // Make pending intents different for each message part
            final int partId = (messageCount <= 1 ? 0 : i + 1);
            if (requireDeliveryReport && (i == (messageCount - 1))) {
                // TODO we only care about the delivery status of the last part
                // Shall we have better tracking of delivery status of all parts?
                deliveryIntents.add(PendingIntent.getBroadcast(
                        context,
                        partId,
                        getSendStatusIntent(context, SendStatusReceiver.MESSAGE_DELIVERED_ACTION,
                                messageUri, partId, subId),
                        0/*flag*/));
            } else {
                deliveryIntents.add(null);
            }
            sentIntents.add(PendingIntent.getBroadcast(
                    context,
                    partId,
                    getSendStatusIntent(context, SendStatusReceiver.MESSAGE_SENT_ACTION,
                            messageUri, partId, subId),
                    0/*flag*/));
        }
        if (sSendMultipartSmsAsSeparateMessages == null) {
            sSendMultipartSmsAsSeparateMessages = MmsConfig.get(subId)
                    .getSendMultipartSmsAsSeparateMessages();
        }
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/SmsSender.java

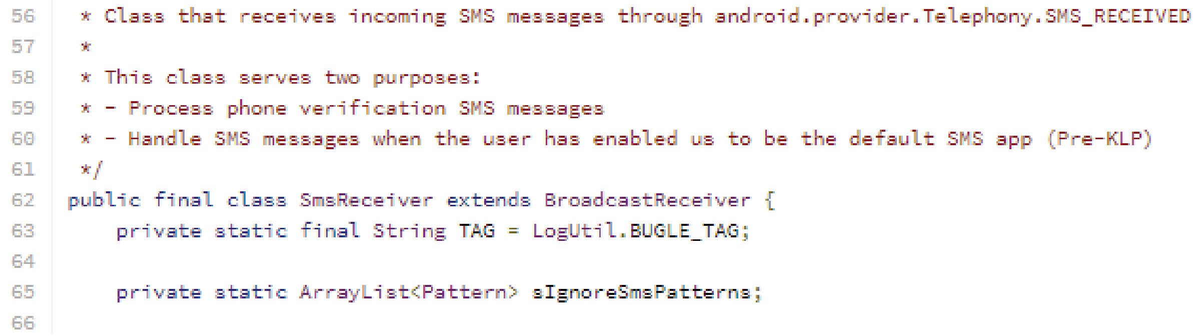
Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>



```
        try {
            if (sSendMultipartSmsAsSeparateMessages) {
                // If multipart sms is not supported, send them as separate messages
                for (int i = 0; i < messageCount; i++) {
                    smsManager.sendTextMessage(dest,
                            serviceCenter,
                            messages.get(i),
                            sentIntents.get(i),
                            deliveryIntents.get(i));
                }
            } else {
                smsManager.sendMultipartTextMessage(
                        dest, serviceCenter, messages, sentIntents, deliveryIntents);
            }
        } catch (final Exception e) {
            throw new SmsException("SmsSender: caught exception in sending " + e);
        }
    }
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/SmsSender.java

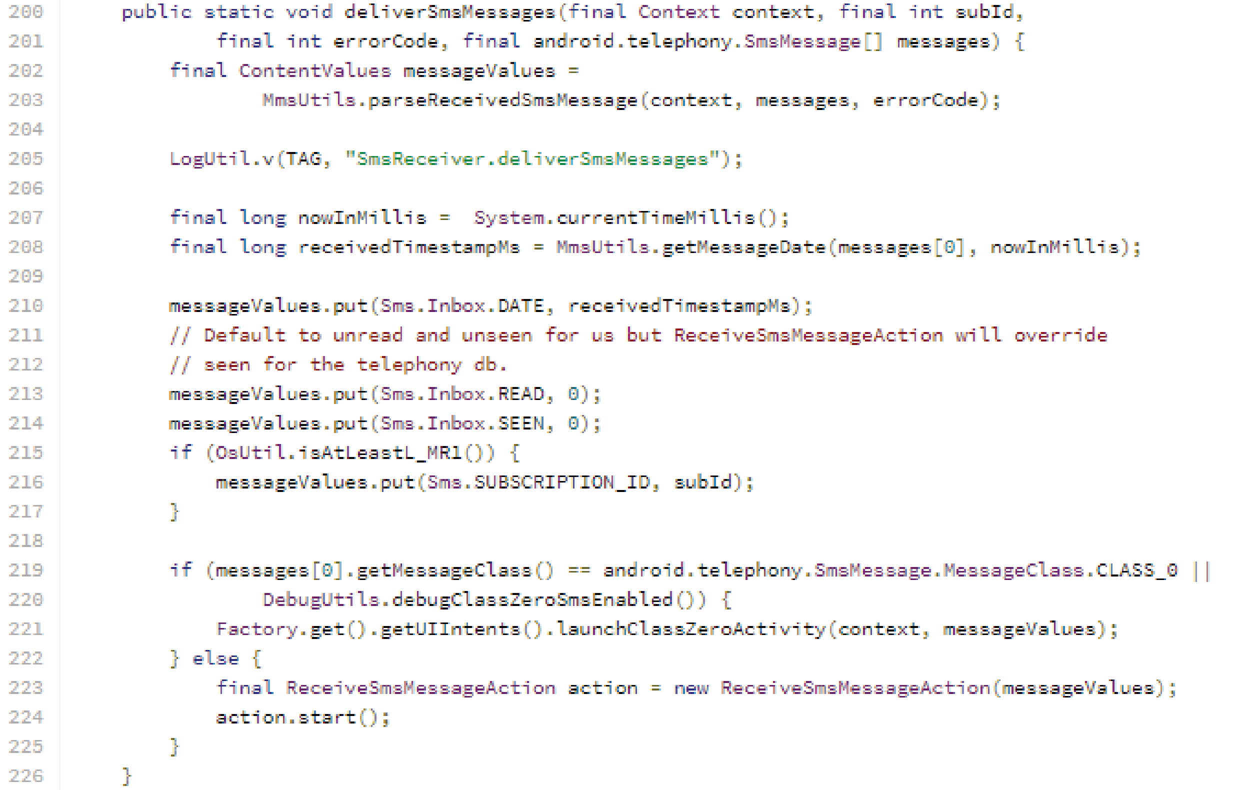


```
 * Class that receives incoming SMS messages through android.provider.Telephony.SMS_RECEIVED
 *
 * This class serves two purposes:
 * - Process phone verification SMS messages
 * - Handle SMS messages when the user has enabled us to be the default SMS app (Pre-KLP)
 */
public final class SmsReceiver extends BroadcastReceiver {
    private static final String TAG = LogUtil.BUGLE_TAG;

    private static ArrayList<Pattern> sIgnoreSmsPatterns;

```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/receiver/SmsReceiver.java

</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>

<pre>
    public static void deliverSmsMessages(final Context context, final int subId,
            final int errorCode, final android.telephony.SmsMessage[] messages) {
        final ContentValues messageValues =
                MmsUtils.parseReceivedSmsMessage(context, messages, errorCode);

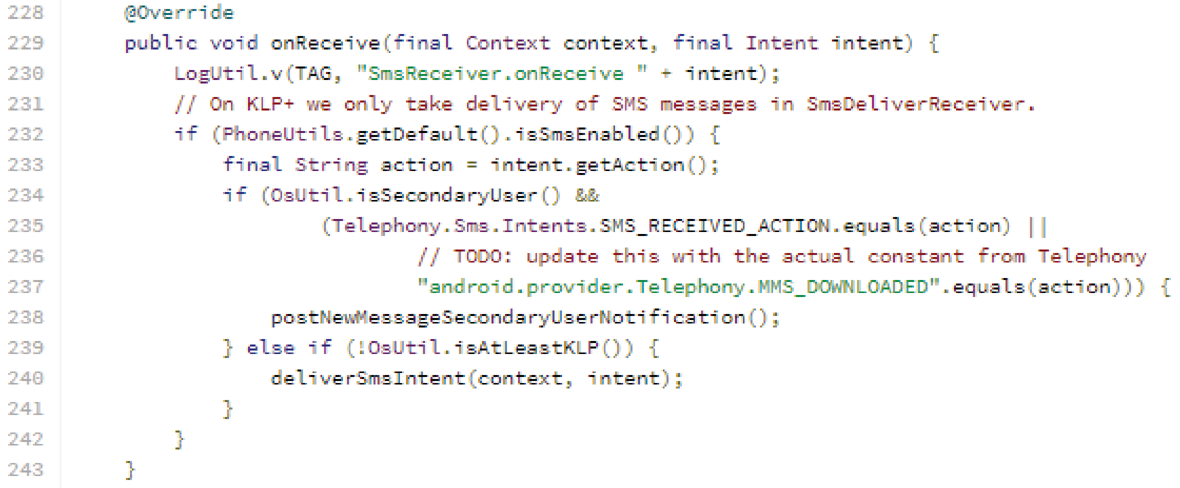
        LogUtil.v(TAG, "SmsReceiver.deliverSmsMessages");

        final long nowInMillis = System.currentTimeMillis();
        final long receivedTimestampMs = MmsUtils.getMessageDate(messages[0], nowInMillis);

        messageValues.put(Sms.Inbox.DATE, receivedTimestampMs);
        // Default to unread and unseen for us but ReceiveSmsMessageAction will override
        // seen for the telephony db.
        messageValues.put(Sms.Inbox.READ, 0);
        messageValues.put(Sms.Inbox.SEEN, 0);
        if (OsUtil.isAtLeastL_MR1()) {
            messageValues.put(Sms.SUBSCRIPTION_ID, subId);
        }

        if (messages[0].getMessageClass() == android.telephony.SmsMessage.MessageClass.CLASS_0 ||
                DebugUtils.debugClassZeroSmsEnabled()) {
            Factory.get().getUIIntents().launchClassZeroActivity(context, messageValues);
        } else {
            final ReceiveSmsMessageAction action = new ReceiveSmsMessageAction(messageValues);
            action.start();
        }
    }
</pre>
https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/receiver/SmsReceiver.java
</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

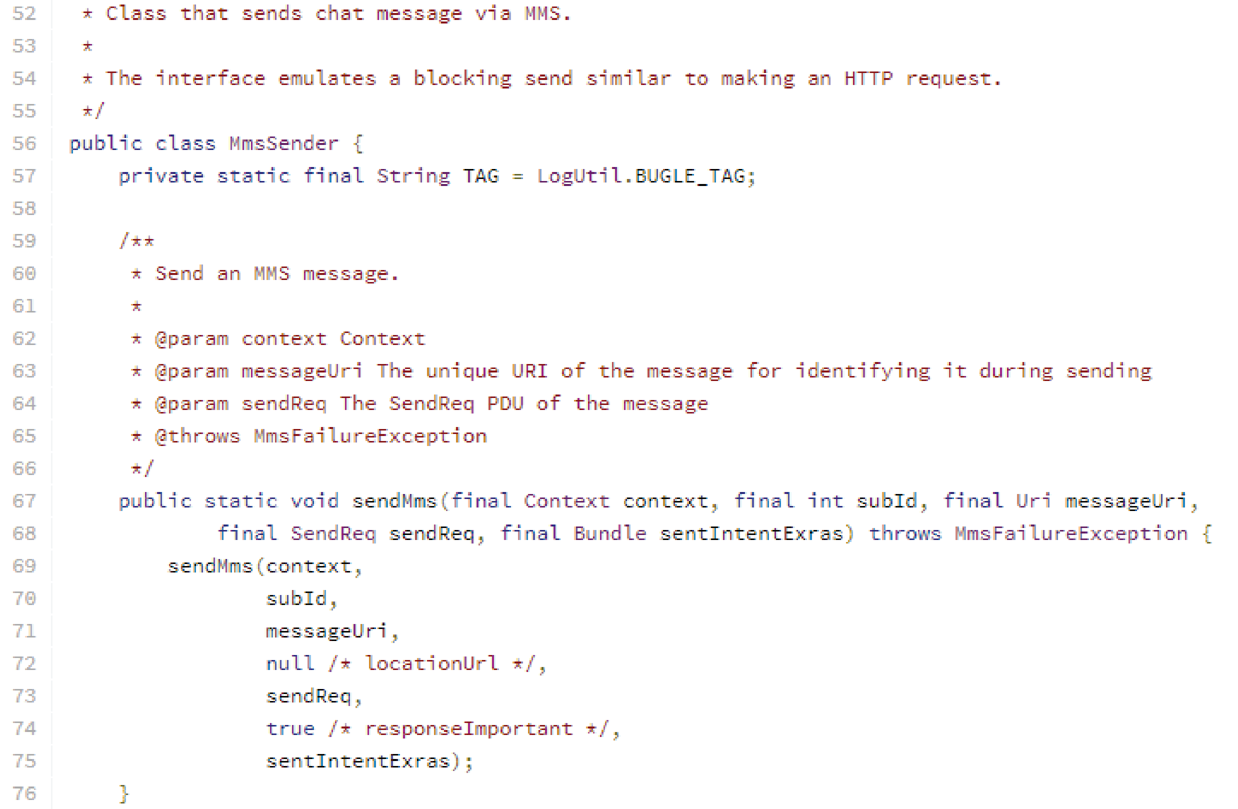
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
@Override
public void onReceive(final Context context, final Intent intent) {
    LogUtil.v(TAG, "SmsReceiver.onReceive " + intent);
    // On KLP+ we only take delivery of SMS messages in SmsDeliverReceiver.
    if (PhoneUtils.getDefault().isSmsEnabled()) {
        final String action = intent.getAction();
        if (OsUtil.isSecondaryUser() &&
                (Telephony.Sms.Intents.SMS_RECEIVED_ACTION.equals(action) ||
                        // TODO: update this with the actual constant from Telephony
                        "android.provider.Telephony.MMS_DOWNLOADED".equals(action))) {
            postNewMessageSecondaryUserNotification();
        } else if (!OsUtil.isAtLeastKLP()) {
            deliverSmsIntent(context, intent);
        }
    }
}
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/receiver/SmsReceiver.java

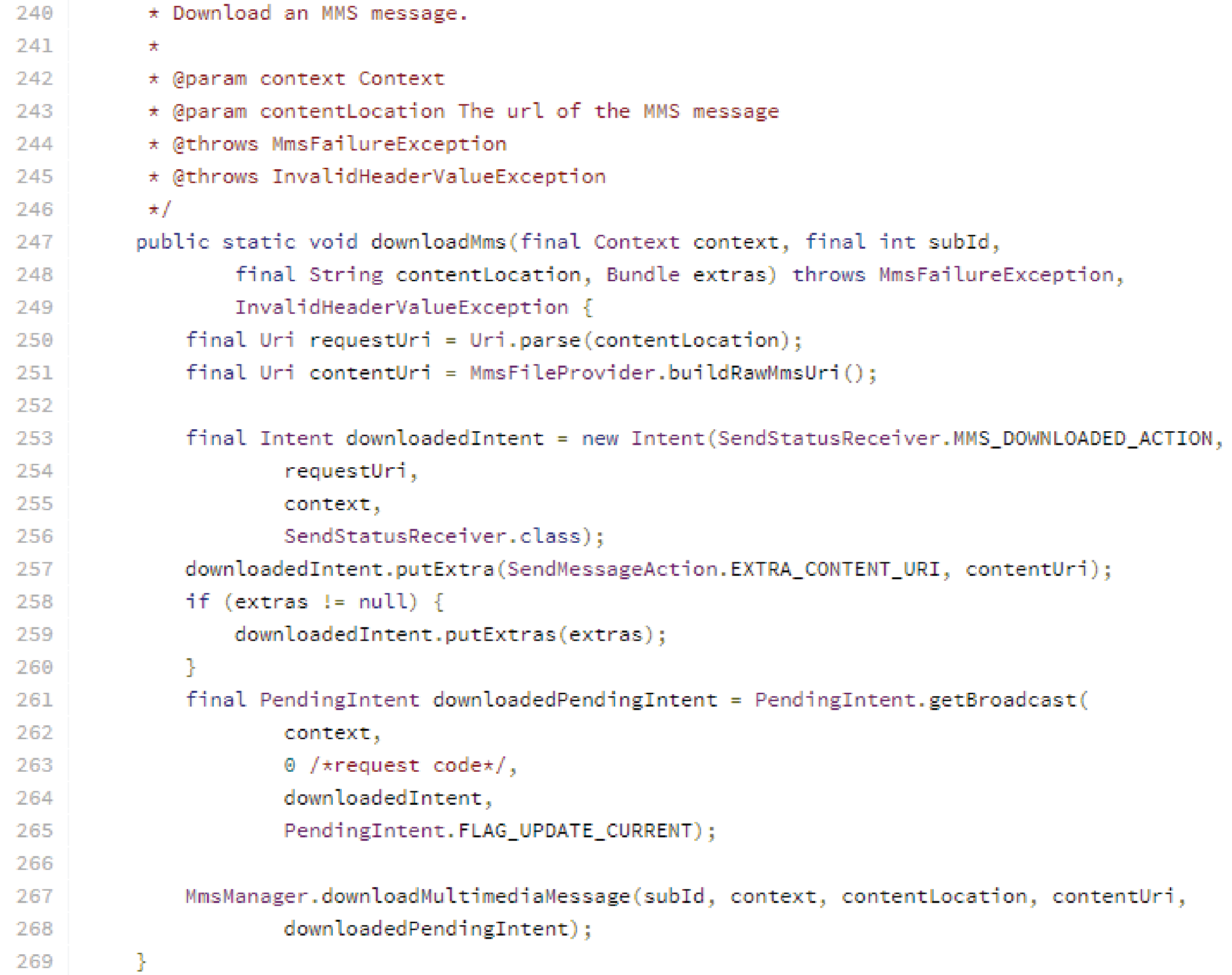
Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products



```
 * Class that sends chat message via MMS.
 *
 * The interface emulates a blocking send similar to making an HTTP request.
 */
public class MmsSender {
    private static final String TAG = LogUtil.BUGLE_TAG;

    /**
     * Send an MMS message.
     *
     * @param context Context
     * @param messageUri The unique URI of the message for identifying it during sending
     * @param sendReq The SendReq PDU of the message
     * @throws MmsFailureException
     */
    public static void sendMms(final Context context, final int subId, final Uri messageUri,
            final SendReq sendReq, final Bundle sentIntentExras) throws MmsFailureException {
        sendMms(context,
                subId,
                messageUri,
                null /* locationUrl */,
                sendReq,
                true /* responseImportant */,
                sentIntentExras);
    }
```

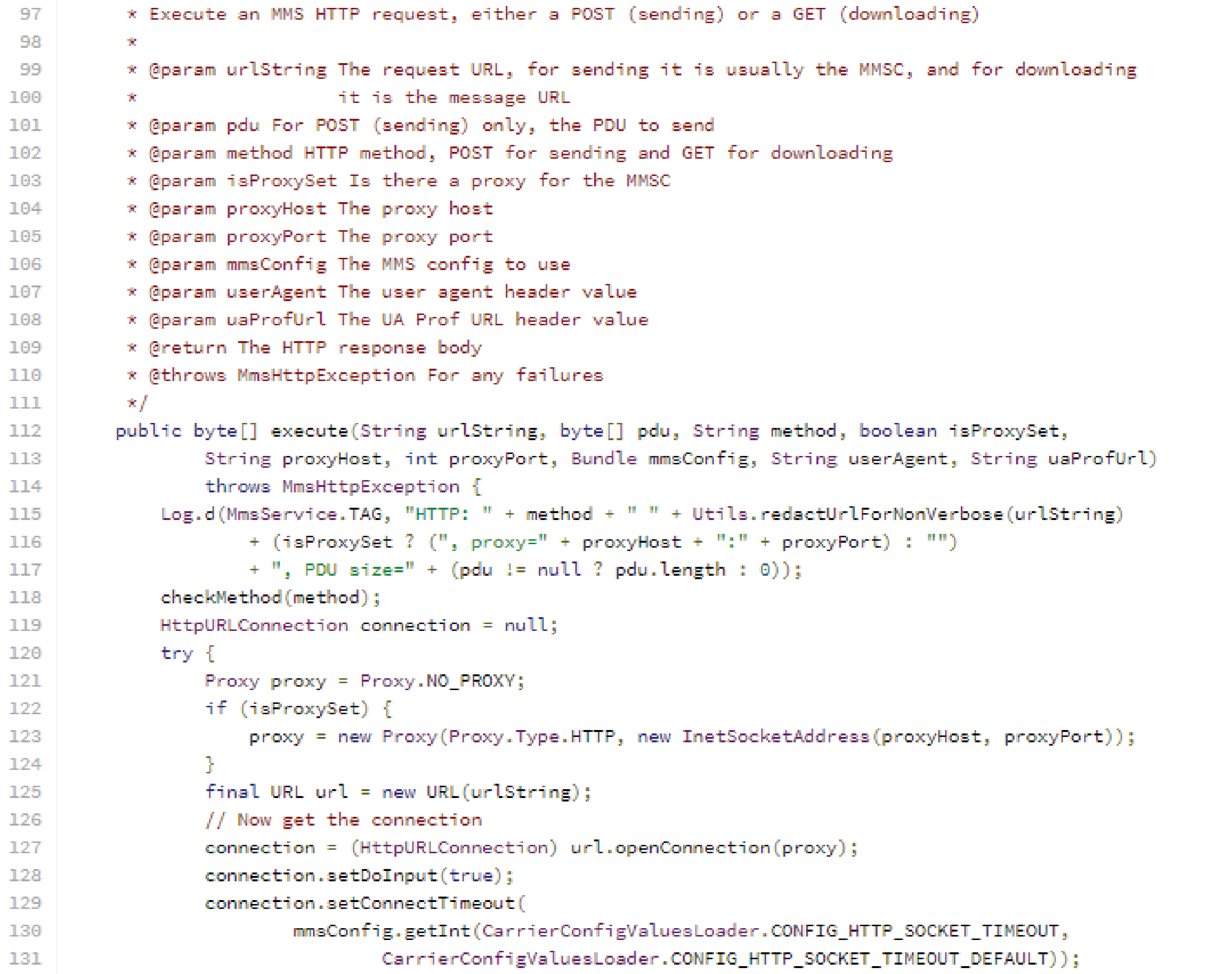
https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/MmsSender.java

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
     * Download an MMS message.
     *
     * @param context Context
     * @param contentLocation The url of the MMS message
     * @throws MmsFailureException
     * @throws InvalidHeaderValueException
     */
    public static void downloadMms(final Context context, final int subId,
            final String contentLocation, Bundle extras) throws MmsFailureException,
            InvalidHeaderValueException {
        final Uri requestUri = Uri.parse(contentLocation);
        final Uri contentUri = MmsFileProvider.buildRawMmsUri();

        final Intent downloadedIntent = new Intent(SendStatusReceiver.MMS_DOWNLOADED_ACTION,
                requestUri,
                context,
                SendStatusReceiver.class);
        downloadedIntent.putExtra(SendMessageAction.EXTRA_CONTENT_URI, contentUri);
        if (extras != null) {
            downloadedIntent.putExtras(extras);
        }
        final PendingIntent downloadedPendingIntent = PendingIntent.getBroadcast(
                context,
                0 /*request code*/,
                downloadedIntent,
                PendingIntent.FLAG_UPDATE_CURRENT);

        MmsManager.downloadMultimediaMessage(subId, context, contentLocation, contentUri,
                downloadedPendingIntent);
    }
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/MmsSender.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
     * Execute an MMS HTTP request, either a POST (sending) or a GET (downloading)
     *
     * @param urlString The request URL, for sending it is usually the MMSC, and for downloading
     *                  it is the message URL
     * @param pdu For POST (sending) only, the PDU to send
     * @param method HTTP method, POST for sending and GET for downloading
     * @param isProxySet Is there a proxy for the MMSC
     * @param proxyHost The proxy host
     * @param proxyPort The proxy port
     * @param mmsConfig The MMS config to use
     * @param userAgent The user agent header value
     * @param uaProfUrl The UA Prof URL header value
     * @return The HTTP response body
     * @throws MmsHttpException For any failures
     */
    public byte[] execute(String urlString, byte[] pdu, String method, boolean isProxySet,
            String proxyHost, int proxyPort, Bundle mmsConfig, String userAgent, String uaProfUrl)
            throws MmsHttpException {
        Log.d(MmsService.TAG, "HTTP: " + method + " " + Utils.redactUrlForNonVerbose(urlString)
                + (isProxySet ? (", proxy=" + proxyHost + ":" + proxyPort) : "")
                + ", PDU size=" + (pdu != null ? pdu.length : 0));
        checkMethod(method);
        HttpURLConnection connection = null;
        try {
            Proxy proxy = Proxy.NO_PROXY;
            if (isProxySet) {
                proxy = new Proxy(Proxy.Type.HTTP, new InetSocketAddress(proxyHost, proxyPort));
            }
            final URL url = new URL(urlString);
            // Now get the connection
            connection = (HttpURLConnection) url.openConnection(proxy);
            connection.setDoInput(true);
            connection.setConnectTimeout(
                    mmsConfig.getInt(CarrierConfigValuesLoader.CONFIG_HTTP_SOCKET_TIMEOUT,
                            CarrierConfigValuesLoader.CONFIG_HTTP_SOCKET_TIMEOUT_DEFAULT));
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/android/support/v7/mms/MmsHttpClient.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

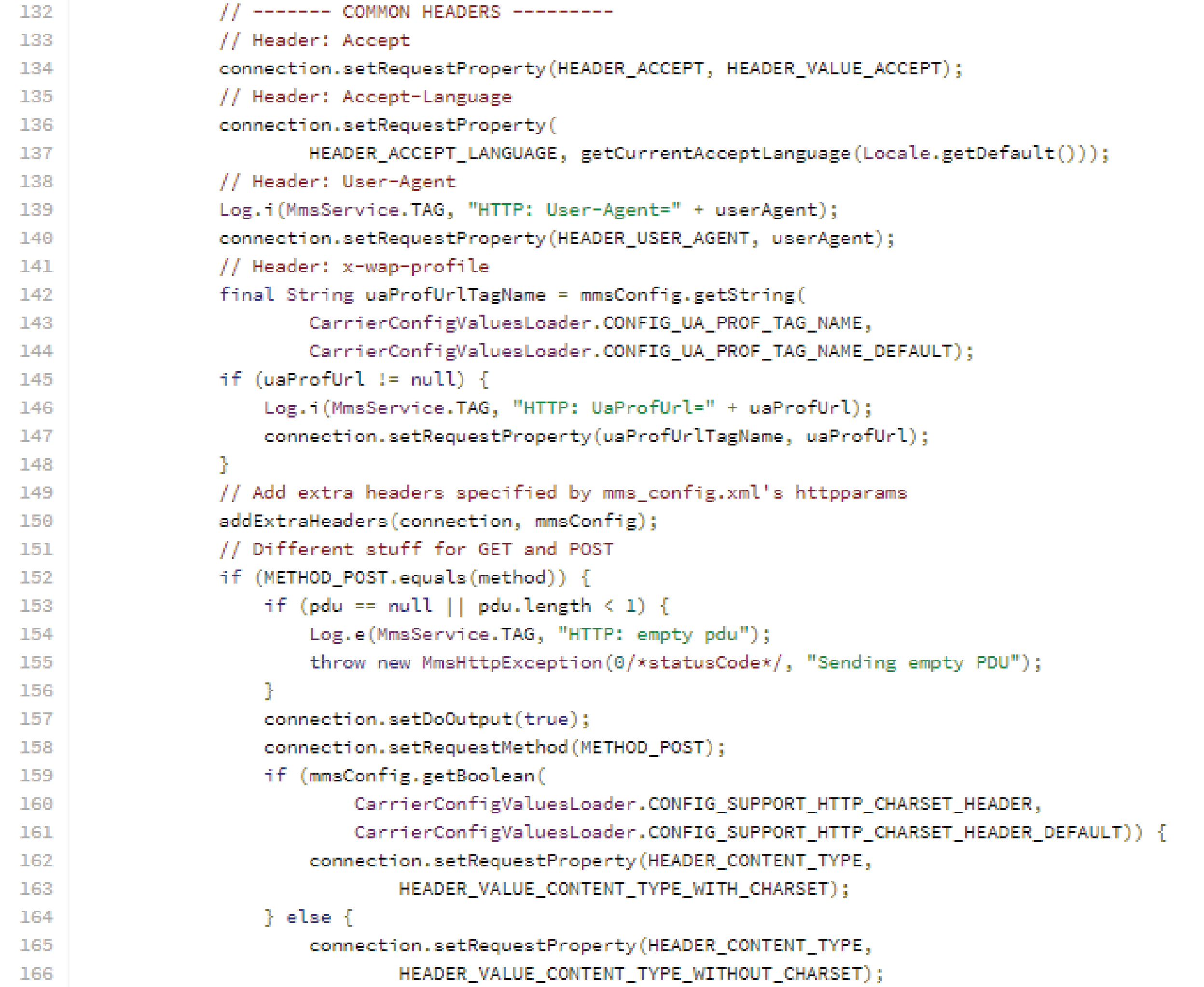
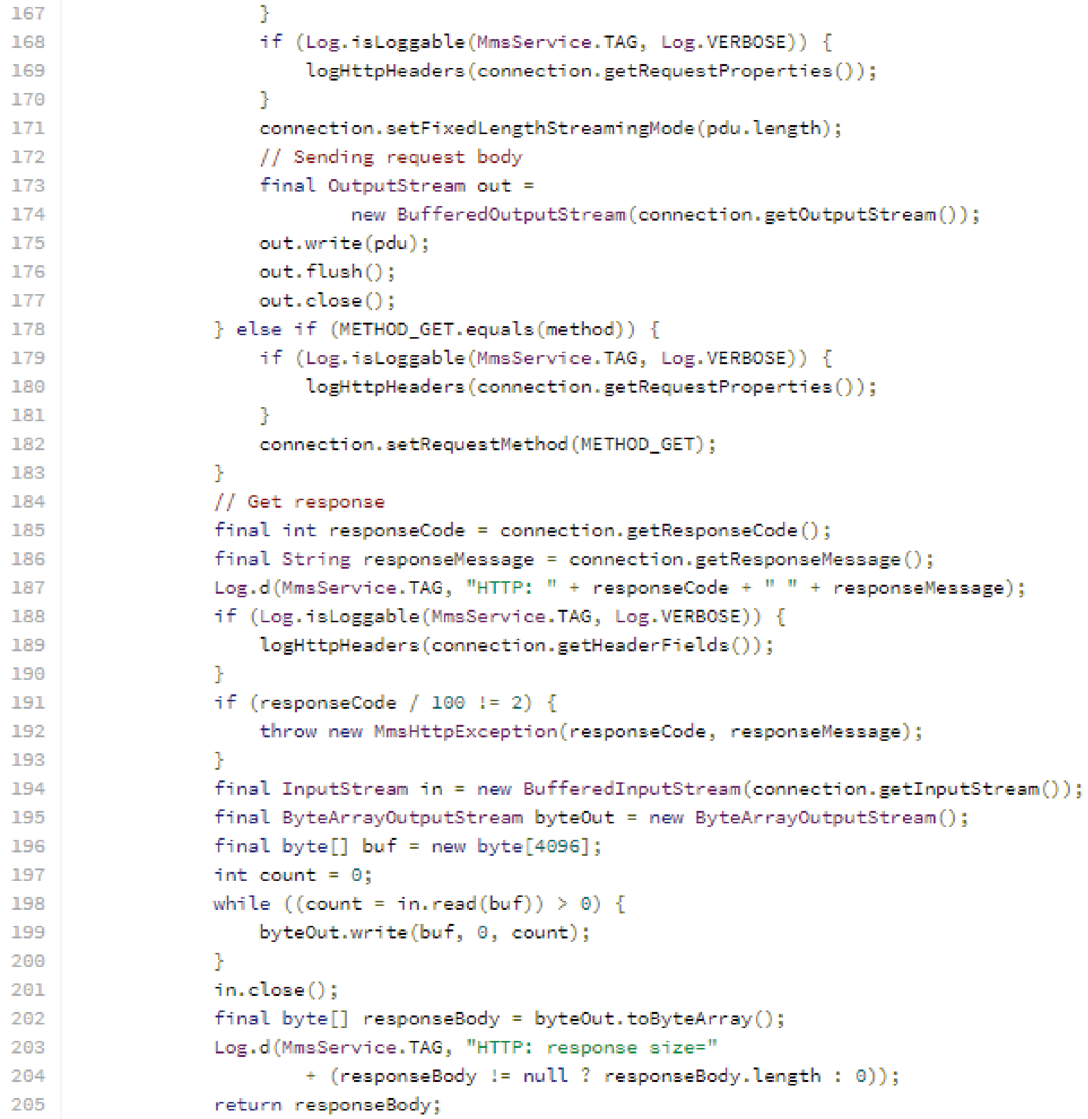
<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>

<pre>
        // ------- COMMON HEADERS ---------
        // Header: Accept
        connection.setRequestProperty(HEADER_ACCEPT, HEADER_VALUE_ACCEPT);
        // Header: Accept-Language
        connection.setRequestProperty(
                HEADER_ACCEPT_LANGUAGE, getCurrentAcceptLanguage(Locale.getDefault()));
        // Header: User-Agent
        Log.i(MmsService.TAG, "HTTP: User-Agent=" + userAgent);
        connection.setRequestProperty(HEADER_USER_AGENT, userAgent);
        // Header: x-wap-profile
        final String uaProfUrlTagName = mmsConfig.getString(
                CarrierConfigValuesLoader.CONFIG_UA_PROF_TAG_NAME,
                CarrierConfigValuesLoader.CONFIG_UA_PROF_TAG_NAME_DEFAULT);
        if (uaProfUrl != null) {
            Log.i(MmsService.TAG, "HTTP: UaProfUrl=" + uaProfUrl);
            connection.setRequestProperty(uaProfUrlTagName, uaProfUrl);
        }
        // Add extra headers specified by mms_config.xml's httpparams
        addExtraHeaders(connection, mmsConfig);
        // Different stuff for GET and POST
        if (METHOD_POST.equals(method)) {
            if (pdu == null || pdu.length < 1) {
                Log.e(MmsService.TAG, "HTTP: empty pdu");
                throw new MmsHttpException(0/*statusCode*/, "Sending empty PDU");
            }
            connection.setDoOutput(true);
            connection.setRequestMethod(METHOD_POST);
            if (mmsConfig.getBoolean(
                    CarrierConfigValuesLoader.CONFIG_SUPPORT_HTTP_CHARSET_HEADER,
                    CarrierConfigValuesLoader.CONFIG_SUPPORT_HTTP_CHARSET_HEADER_DEFAULT)) {
                connection.setRequestProperty(HEADER_CONTENT_TYPE,
                        HEADER_VALUE_CONTENT_TYPE_WITH_CHARSET);
            } else {
                connection.setRequestProperty(HEADER_CONTENT_TYPE,
                        HEADER_VALUE_CONTENT_TYPE_WITHOUT_CHARSET);
</pre>
https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/android/support/v7/mms/MmsHttpClient.java
</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

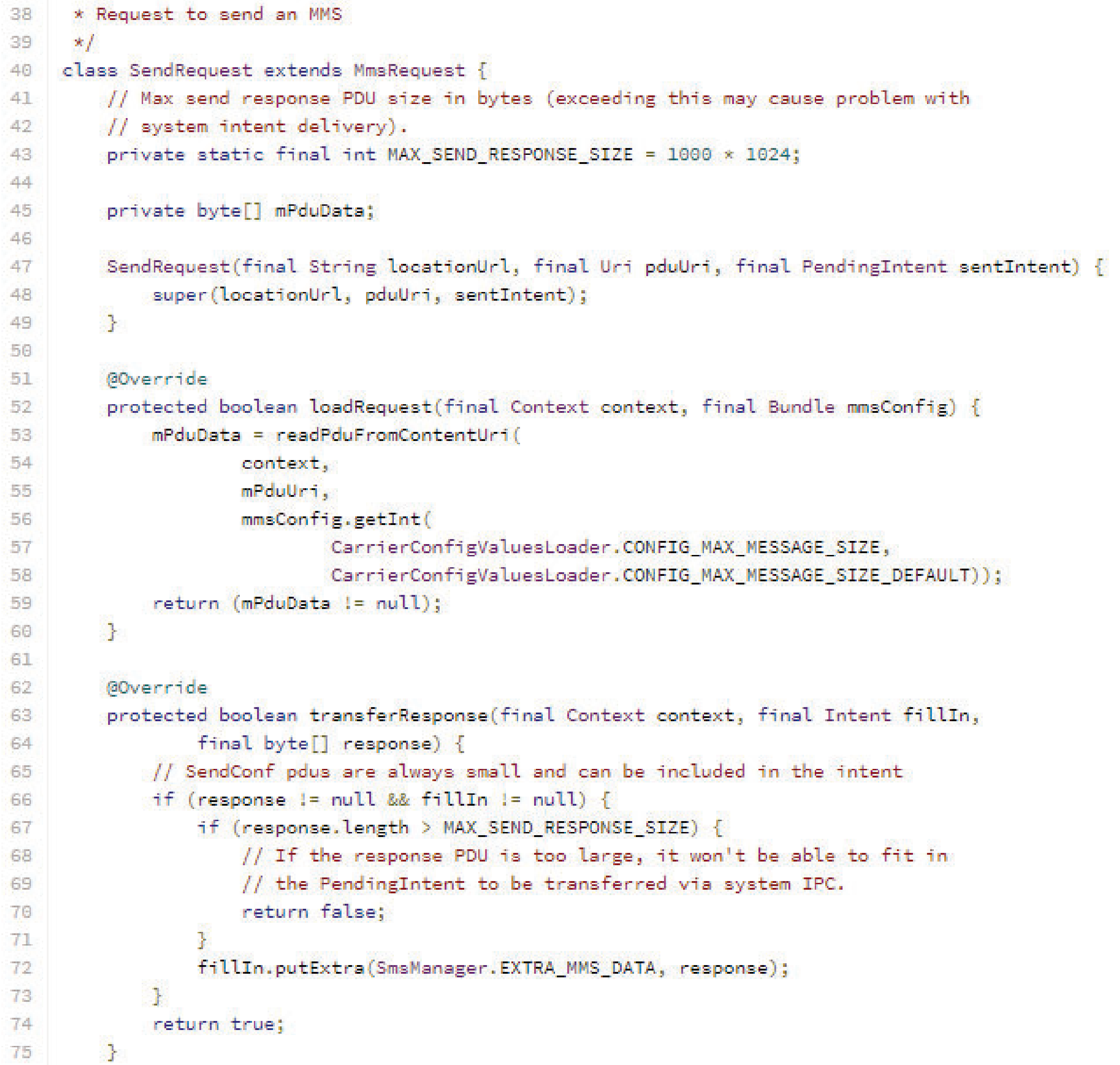
US9467838	Exemplary Supporting Evidence Regarding Accused Products



```
            }
            if (Log.isLoggable(MmsService.TAG, Log.VERBOSE)) {
                logHttpHeaders(connection.getRequestProperties());
            }
            connection.setFixedLengthStreamingMode(pdu.length);
            // Sending request body
            final OutputStream out =
                    new BufferedOutputStream(connection.getOutputStream());
            out.write(pdu);
            out.flush();
            out.close();
        } else if (METHOD_GET.equals(method)) {
            if (Log.isLoggable(MmsService.TAG, Log.VERBOSE)) {
                logHttpHeaders(connection.getRequestProperties());
            }
            connection.setRequestMethod(METHOD_GET);
        }
        // Get response
        final int responseCode = connection.getResponseCode();
        final String responseMessage = connection.getResponseMessage();
        Log.d(MmsService.TAG, "HTTP: " + responseCode + " " + responseMessage);
        if (Log.isLoggable(MmsService.TAG, Log.VERBOSE)) {
            logHttpHeaders(connection.getHeaderFields());
        }
        if (responseCode / 100 != 2) {
            throw new MmsHttpException(responseCode, responseMessage);
        }
        final InputStream in = new BufferedInputStream(connection.getInputStream());
        final ByteArrayOutputStream byteOut = new ByteArrayOutputStream();
        final byte[] buf = new byte[4096];
        int count = 0;
        while ((count = in.read(buf)) > 0) {
            byteOut.write(buf, 0, count);
        }
        in.close();
        final byte[] responseBody = byteOut.toByteArray();
        Log.d(MmsService.TAG, "HTTP: response size="
                + (responseBody != null ? responseBody.length : 0));
        return responseBody;
```

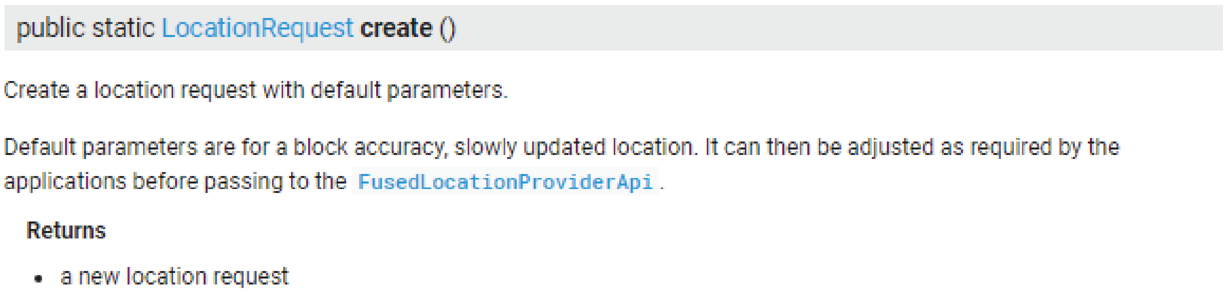
https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/android/support/v7/mms/MmsHttpClient.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
 * Request to send an MMS
 */
class SendRequest extends MmsRequest {
    // Max send response PDU size in bytes (exceeding this may cause problem with
    // system intent delivery).
    private static final int MAX_SEND_RESPONSE_SIZE = 1000 * 1024;

    private byte[] mPduData;

    SendRequest(final String locationUrl, final Uri pduUri, final PendingIntent sentIntent) {
        super(locationUrl, pduUri, sentIntent);
    }

    @Override
    protected boolean loadRequest(final Context context, final Bundle mmsConfig) {
        mPduData = readPduFromContentUri(
                context,
                mPduUri,
                mmsConfig.getInt(
                        CarrierConfigValuesLoader.CONFIG_MAX_MESSAGE_SIZE,
                        CarrierConfigValuesLoader.CONFIG_MAX_MESSAGE_SIZE_DEFAULT));
        return (mPduData != null);
    }

    @Override
    protected boolean transferResponse(final Context context, final Intent fillIn,
            final byte[] response) {
        // SendConf pdus are always small and can be included in the intent
        if (response != null && fillIn != null) {
            if (response.length > MAX_SEND_RESPONSE_SIZE) {
                // If the response PDU is too large, it won't be able to fit in
                // the PendingIntent to be transferred via system IPC.
                return false;
            }
            fillIn.putExtra(SmsManager.EXTRA_MMS_DATA, response);
        }
        return true;
    }
```

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/android/support/v7/mms/SendRequest.java  public static LocationRequest **create** () Create a location request with default parameters. Default parameters are for a block accuracy, slowly updated location. It can then be adjusted as required by the applications before passing to the `FusedLocationProviderApi` . **Returns** - a new location request https://developers.google.com/android/reference/com/google/android/gms/location/LocationRequest

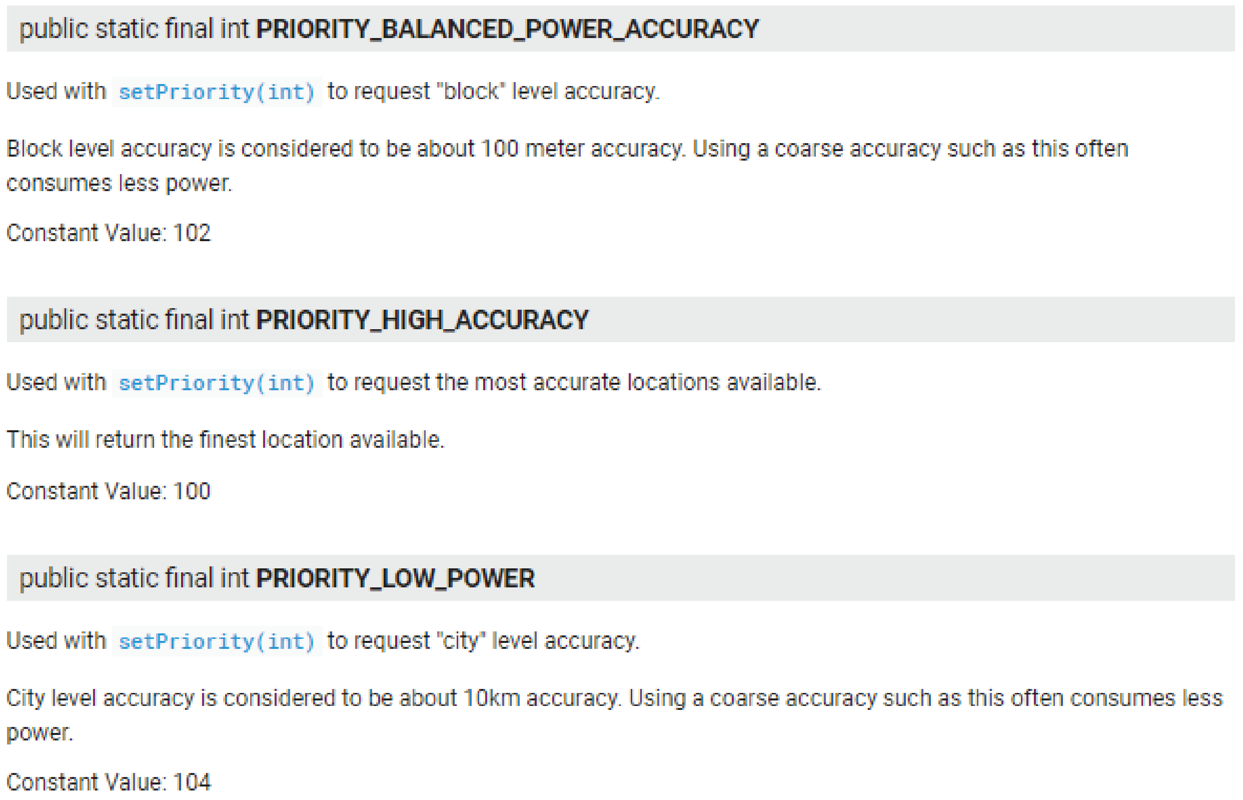
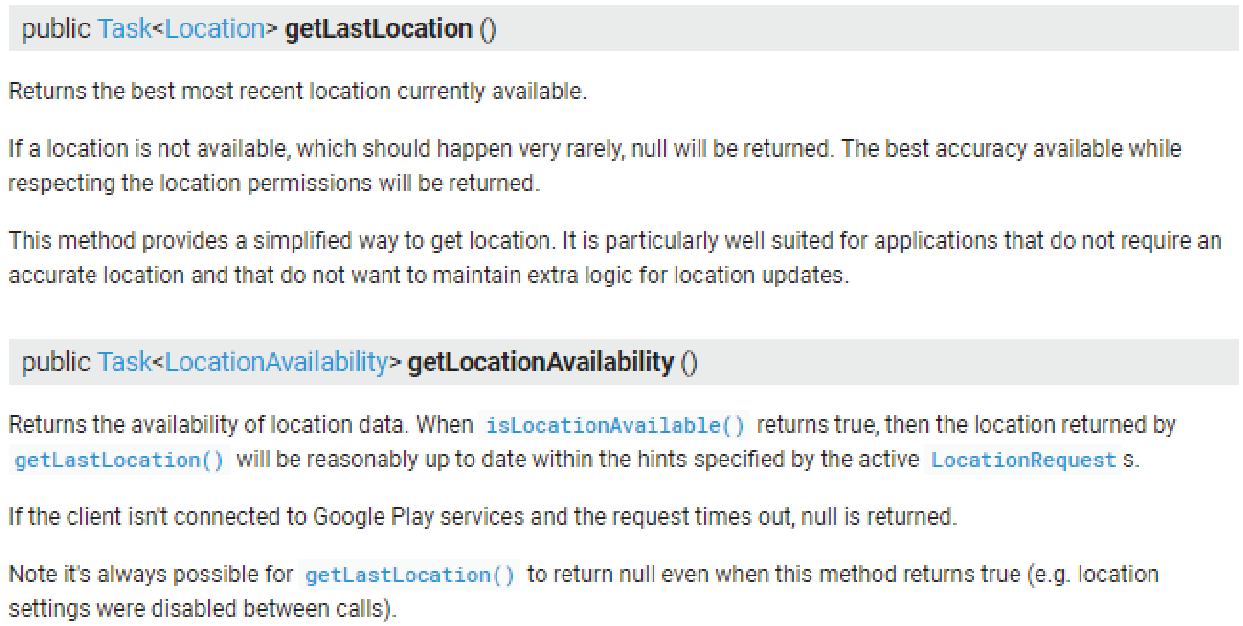
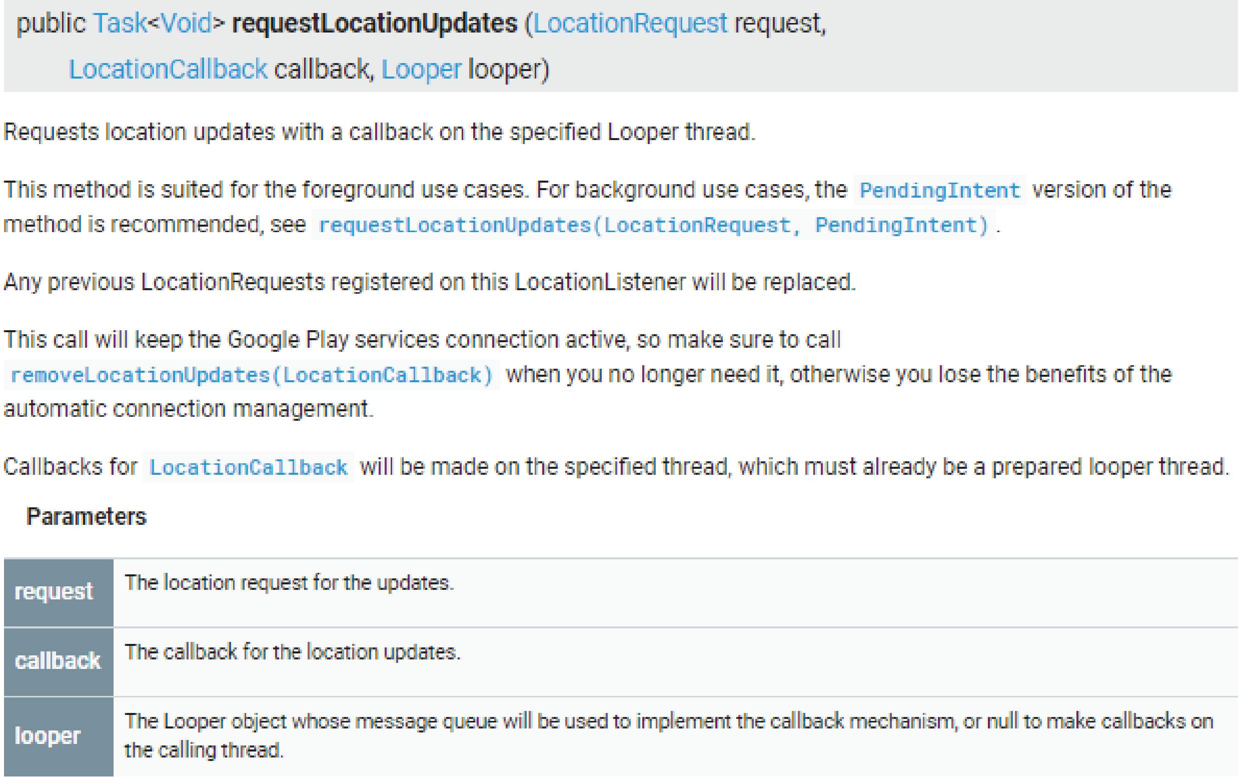
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public static final int **PRIORITY_BALANCED_POWER_ACCURACY** Used with `setPriority(int)` to request "block" level accuracy. Block level accuracy is considered to be about 100 meter accuracy. Using a coarse accuracy such as this often consumes less power. Constant Value: 102 public static final int **PRIORITY_HIGH_ACCURACY** Used with `setPriority(int)` to request the most accurate locations available. This will return the finest location available. Constant Value: 100 public static final int **PRIORITY_LOW_POWER** Used with `setPriority(int)` to request "city" level accuracy. City level accuracy is considered to be about 10km accuracy. Using a coarse accuracy such as this often consumes less power. Constant Value: 104 https://developers.google.com/android/reference/com/google/android/gms/location/LocationRequest

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public Task<Location> **getLastLocation** () Returns the best most recent location currently available. If a location is not available, which should happen very rarely, null will be returned. The best accuracy available while respecting the location permissions will be returned. This method provides a simplified way to get location. It is particularly well suited for applications that do not require an accurate location and that do not want to maintain extra logic for location updates. public Task<LocationAvailability> **getLocationAvailability** () Returns the availability of location data. When `isLocationAvailable()` returns true, then the location returned by `getLastLocation()` will be reasonably up to date within the hints specified by the active `LocationRequest` s. If the client isn't connected to Google Play services and the request times out, null is returned. Note it's always possible for `getLastLocation()` to return null even when this method returns true (e.g. location settings were disabled between calls). https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>


public Task<Void> **requestLocationUpdates** (LocationRequest request, LocationCallback callback, Looper looper)

Requests location updates with a callback on the specified Looper thread.

This method is suited for the foreground use cases. For background use cases, the `PendingIntent` version of the method is recommended, see `requestLocationUpdates(LocationRequest, PendingIntent)`.

Any previous LocationRequests registered on this LocationListener will be replaced.

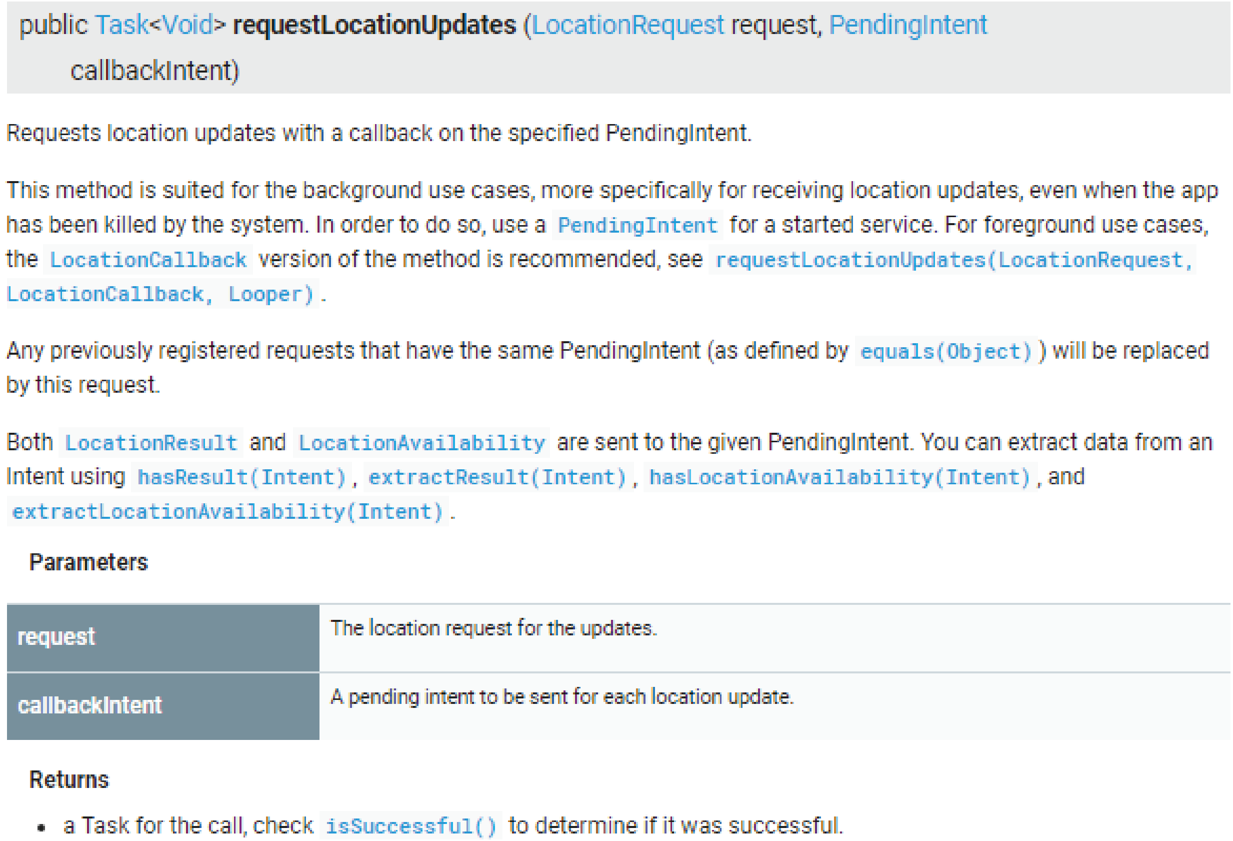
This call will keep the Google Play services connection active, so make sure to call `removeLocationUpdates(LocationCallback)` when you no longer need it, otherwise you lose the benefits of the automatic connection management.

Callbacks for `LocationCallback` will be made on the specified thread, which must already be a prepared looper thread.

Parameters

<table>
<tr><td><b>request</b></td><td>The location request for the updates.</td></tr>
<tr><td><b>callback</b></td><td>The callback for the location updates.</td></tr>
<tr><td><b>looper</b></td><td>The Looper object whose message queue will be used to implement the callback mechanism, or null to make callbacks on the calling thread.</td></tr>
</table>

https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient
</td></tr>
</table>

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

public Task<Void> **requestLocationUpdates** (LocationRequest request, PendingIntent callbackIntent)

Requests location updates with a callback on the specified PendingIntent.

This method is suited for the background use cases, more specifically for receiving location updates, even when the app has been killed by the system. In order to do so, use a `PendingIntent` for a started service. For foreground use cases, the `LocationCallback` version of the method is recommended, see `requestLocationUpdates(LocationRequest, LocationCallback, Looper)` .

Any previously registered requests that have the same PendingIntent (as defined by `equals(Object)`) will be replaced by this request.

Both `LocationResult` and `LocationAvailability` are sent to the given PendingIntent. You can extract data from an Intent using `hasResult(Intent)` , `extractResult(Intent)` , `hasLocationAvailability(Intent)` , and `extractLocationAvailability(Intent)` .

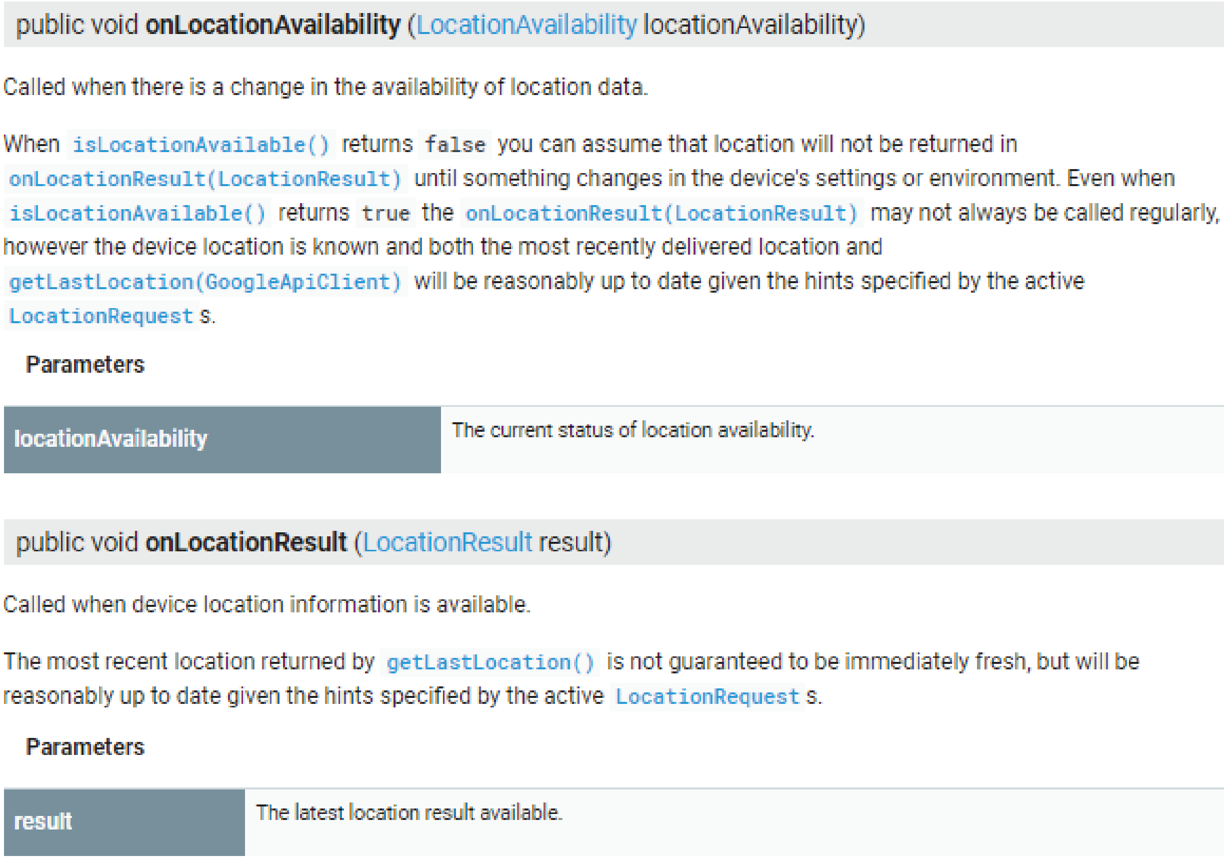
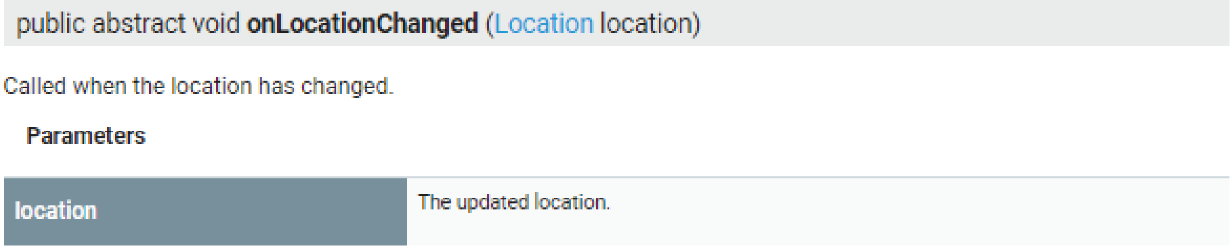
Parameters

request	The location request for the updates.
callbackIntent	A pending intent to be sent for each location update.

Returns

- a Task for the call, check `isSuccessful()` to determine if it was successful.

https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public void **onLocationAvailability** (LocationAvailability locationAvailability) Called when there is a change in the availability of location data. When `isLocationAvailable()` returns `false` you can assume that location will not be returned in `onLocationResult(LocationResult)` until something changes in the device's settings or environment. Even when `isLocationAvailable()` returns `true` the `onLocationResult(LocationResult)` may not always be called regularly, however the device location is known and both the most recently delivered location and `getLastLocation(GoogleApiClient)` will be reasonably up to date given the hints specified by the active `LocationRequest` s. **Parameters** **locationAvailability** The current status of location availability. public void **onLocationResult** (LocationResult result) Called when device location information is available. The most recent location returned by `getLastLocation()` is not guaranteed to be immediately fresh, but will be reasonably up to date given the hints specified by the active `LocationRequest` s. **Parameters** **result** The latest location result available. https://developers.google.com/android/reference/com/google/android/gms/location/LocationCallback  public abstract void **onLocationChanged** (Location location) Called when the location has changed. **Parameters** **location** The updated location.

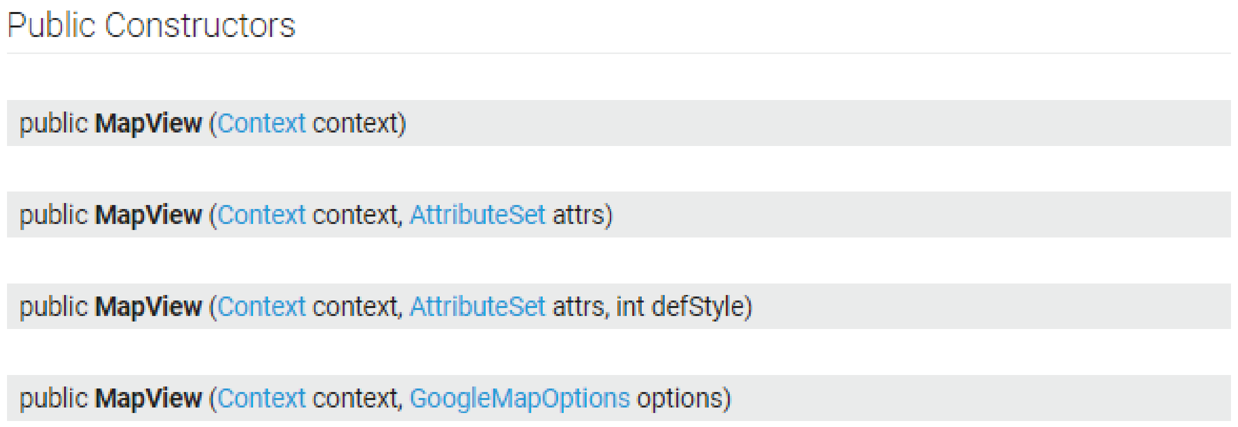
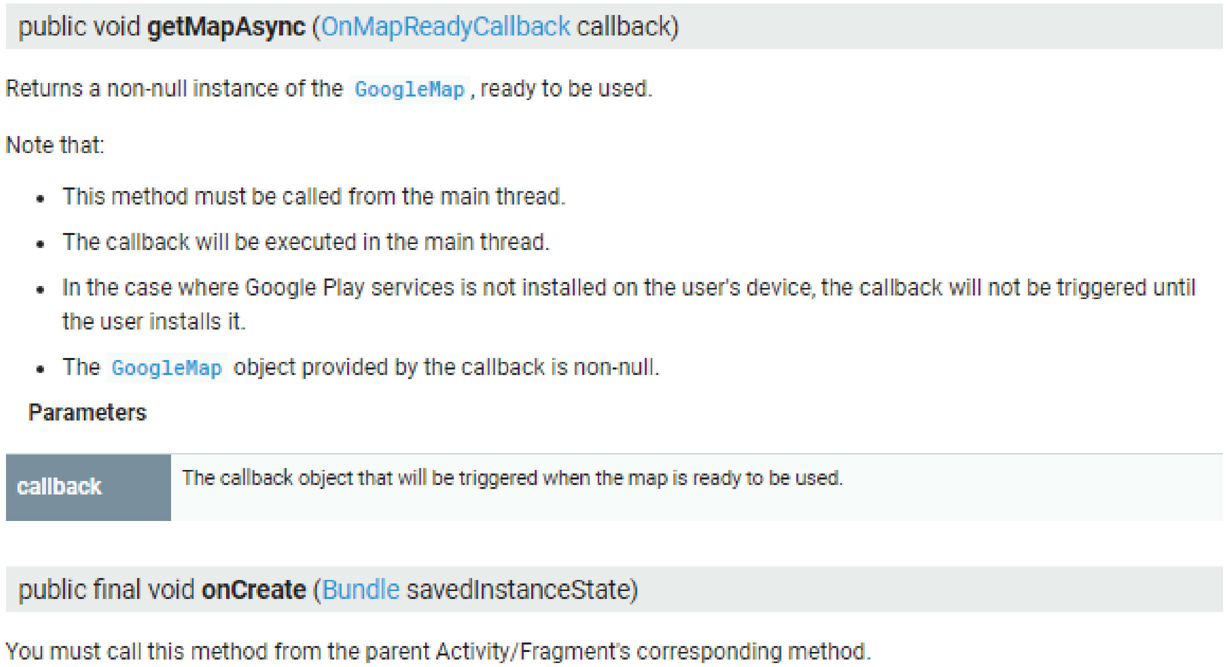
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	https://developers.google.com/android/reference/com/google/android/gms/location/LocationListener  Public Constructors public **MapView** (Context context) public **MapView** (Context context, AttributeSet attrs) public **MapView** (Context context, AttributeSet attrs, int defStyle) public **MapView** (Context context, GoogleMapOptions options) https://developers.google.com/android/reference/com/google/android/gms/maps/MapView

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public void **getMapAsync** (OnMapReadyCallback callback) Returns a non-null instance of the `GoogleMap`, ready to be used. Note that: • This method must be called from the main thread. • The callback will be executed in the main thread. • In the case where Google Play services is not installed on the user's device, the callback will not be triggered until the user installs it. • The `GoogleMap` object provided by the callback is non-null. **Parameters** callback — The callback object that will be triggered when the map is ready to be used. public final void **onCreate** (Bundle savedInstanceState) You must call this method from the parent Activity/Fragment's corresponding method. https://developers.google.com/android/reference/com/google/android/gms/maps/MapView
[1F] receiving, from the second server, the second georeferenced map data;	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of: receiving, from the second server, the second georeferenced map data. **Regarding Find My Device** and Android Device Manager, the user, via the first device, or the device itself receives second map data from a second server. The received second map data occurs responsive to user input (e.g., zoom, drag, pan, change focus, change map type, refresh or reload request device or symbol selection, another device or user selection, change in position of first device, change in position of a second device). Alternatively, the second may data may be received responsive to an automatic and/or pre-determined control caused by an instruction from within the first device or from the one or more second devices, e.g. a refresh. The received second map data includes an update to the first data or a

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	replacement of the first data. The second map data may come from one of many sources and servers. The second map data includes georeferenced data. **<u>Regarding Google Maps</u>**, Google Latitude, Google Plus, Google Hangouts, Google Messages, Google Allo, Google Duo, Google Chrome, and Android Messenger, the user, via the first device, or the device itself receives second map data from a second server. The received second map data occurs responsive to user input (e.g., zoom, drag, pan, change focus, refresh or reload request, change map type, device or symbol selection, another device or user selection, change in position of first device, change in position of a second device). Alternatively, the second may data may be received responsive to an automatic and/or pre-determined control caused by an instruction from within the first device or from the one or more second devices, e.g. a refresh. The received second map data includes an update to the first data or a replacement of the first data. The second map data may come from one of many sources and servers. The second map data includes georeferenced data. **<u>Exemplary Support for Find My Device</u>**:

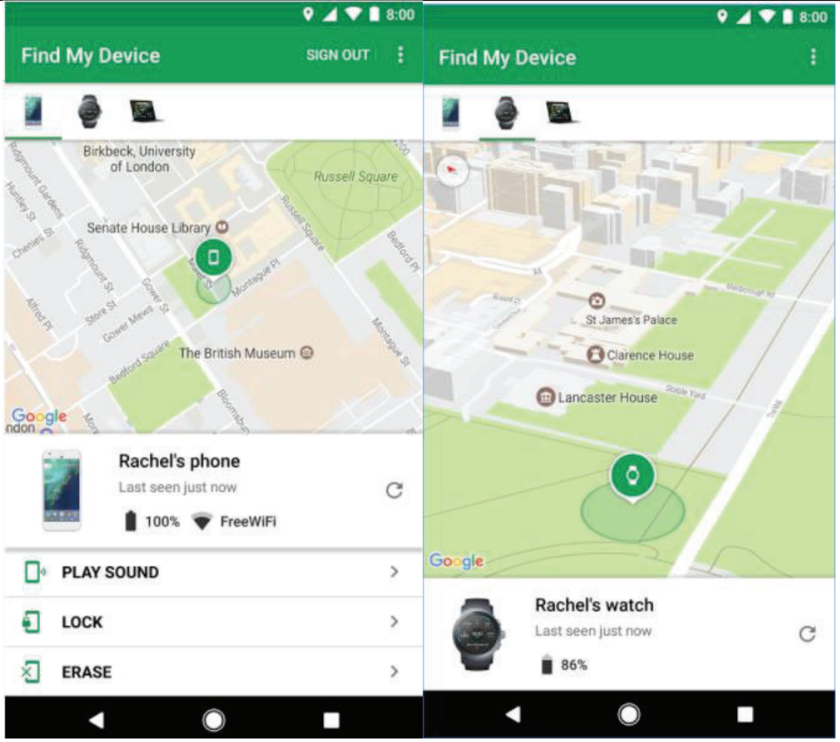
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 8:00 Find My Device SIGN OUT Birkbeck, University of London Russell Square Senate House Library The British Museum Google Rachel's phone Last seen just now 100% FreeWiFi PLAY SOUND LOCK ERASE 8:00 Find My Device St James's Palace Clarence House Lancaster House Google Rachel's watch Last seen just now 86% https://play.google.com/store/apps/details?id=com.google.android.apps.adm&hl=en

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

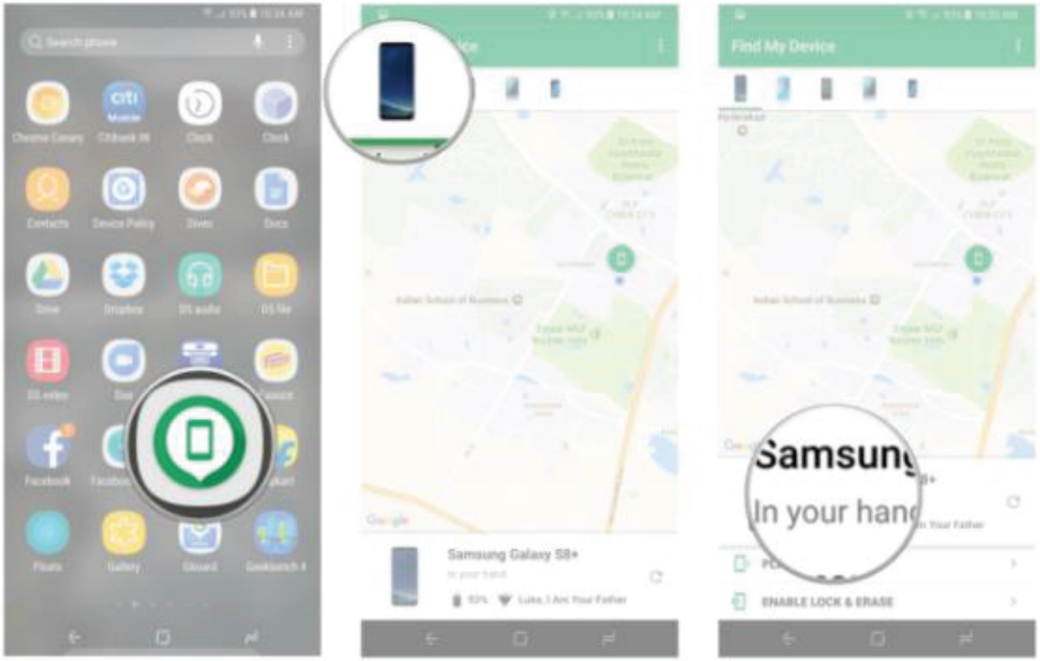
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	Once you're signed in to Find My Device, you'll see a map with your current location as well as the make and model of your phone, and two options — Play Sound, and Enable Lock & Erase. Hitting the latter option will allow you to start using the Lock and Erase functions. If you've signed into more than one phone, you can select a particular device by browsing the list at the top of the screen. 1. Open **Find My Device** from your home screen or app drawer. 2. Select your phone from the **list of devices at the top of the screen**. 3. See if your phone is **discoverable**.  Search phone Find My Device Samsung Galaxy S8+ Samsung In your hand ENABLE LOCK & ERASE https://www.androidcentral.com/find-my-device

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

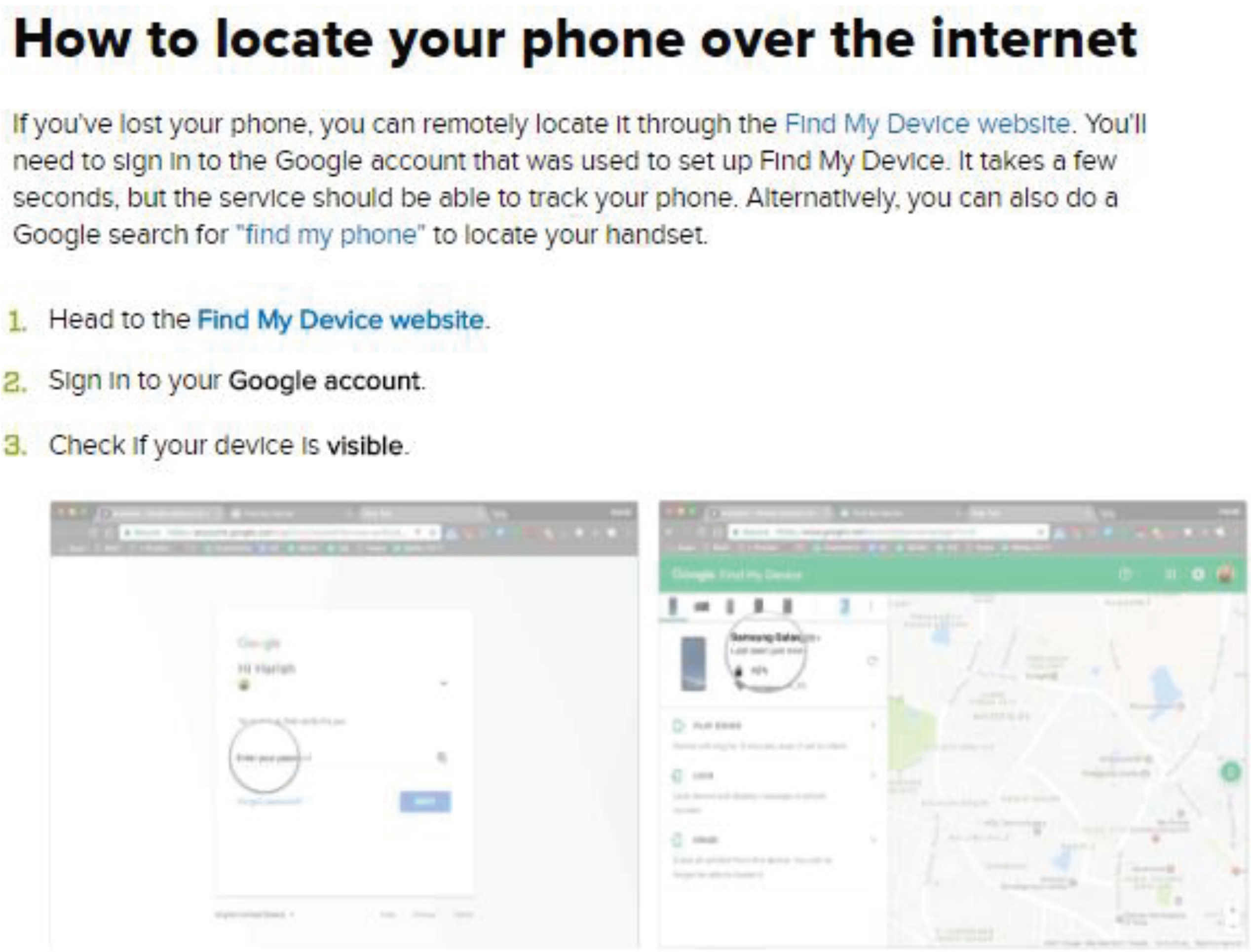
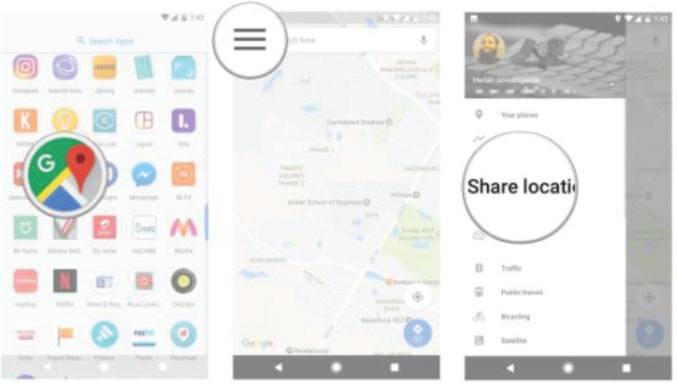
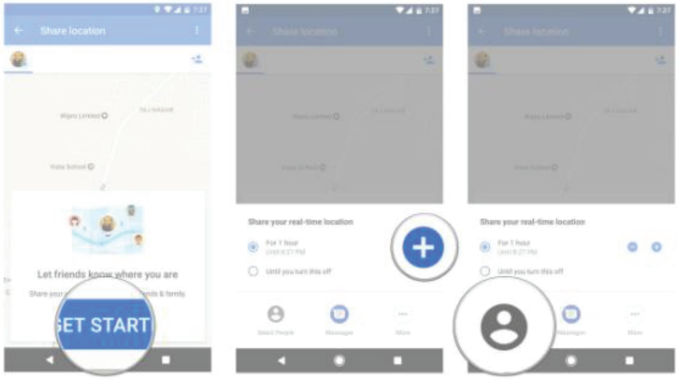
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 **How to locate your phone over the internet** If you've lost your phone, you can remotely locate it through the Find My Device website. You'll need to sign in to the Google account that was used to set up Find My Device. It takes a few seconds, but the service should be able to track your phone. Alternatively, you can also do a Google search for "find my phone" to locate your handset. 1. Head to the **Find My Device website**. 2. Sign in to your **Google account**. 3. Check if your device is **visible**. https://www.androidcentral.com/find-my-device **<u>Exemplary Support for Google Maps</u>**:

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to share your location in Google Maps 1. Open **Google Maps** from the app drawer or the home screen. 2. Tap the **hamburger menu** (the three horizontal lines) on the top left corner of the screen. 3. Select **Share location**.  4. Tap **Get Started**. 5. Use the **+** icon to select a time period or select the **Until you turn this off** setting to share your location indefinitely. 6. Tap **Select People**.  Share location Let friends know where you are ET START Share your real-time location For 1 hour Until you turn this off https://www.androidcentral.com/how-share-location-google-maps

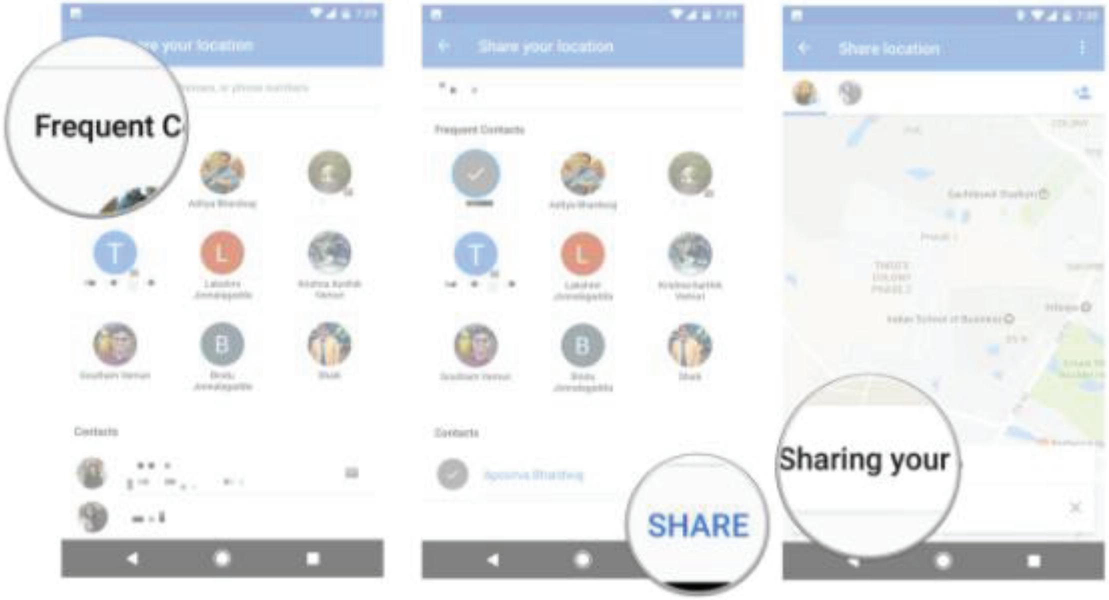
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	7. You'll see a list of your frequent contacts at the top, along with a full list of contacts. **Pick the contacts** by tapping their name. 8. Once you've selected the contacts you want to share your location to, tap **Share**. 9. You'll see a message saying that the selected contact can view your location.  Frequent C Share your location Frequent Contacts Contacts SHARE Share location Sharing your https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

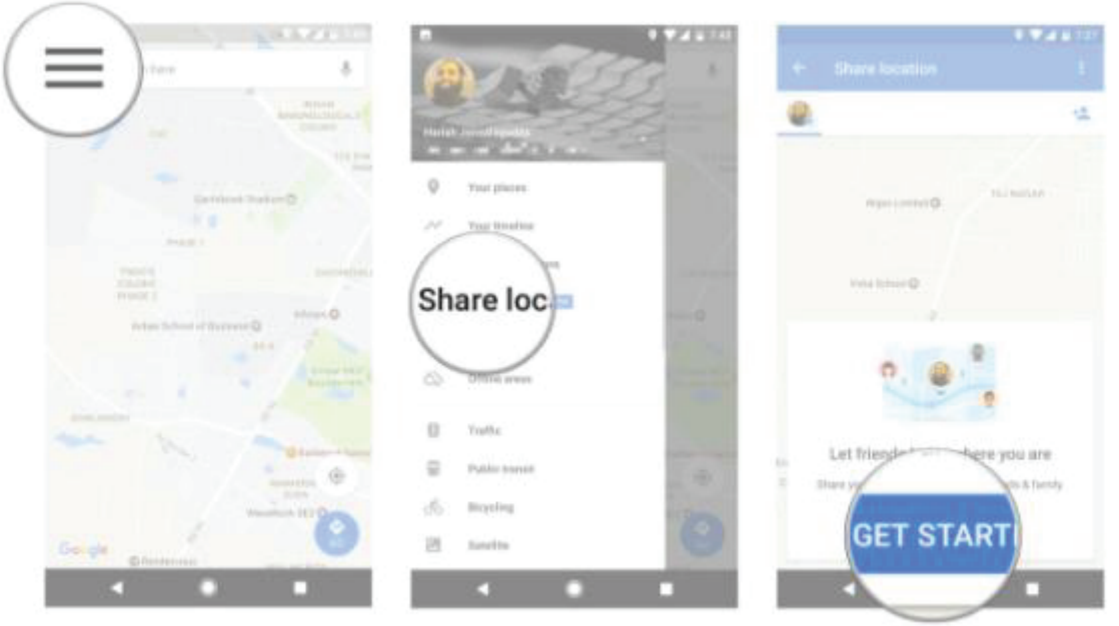
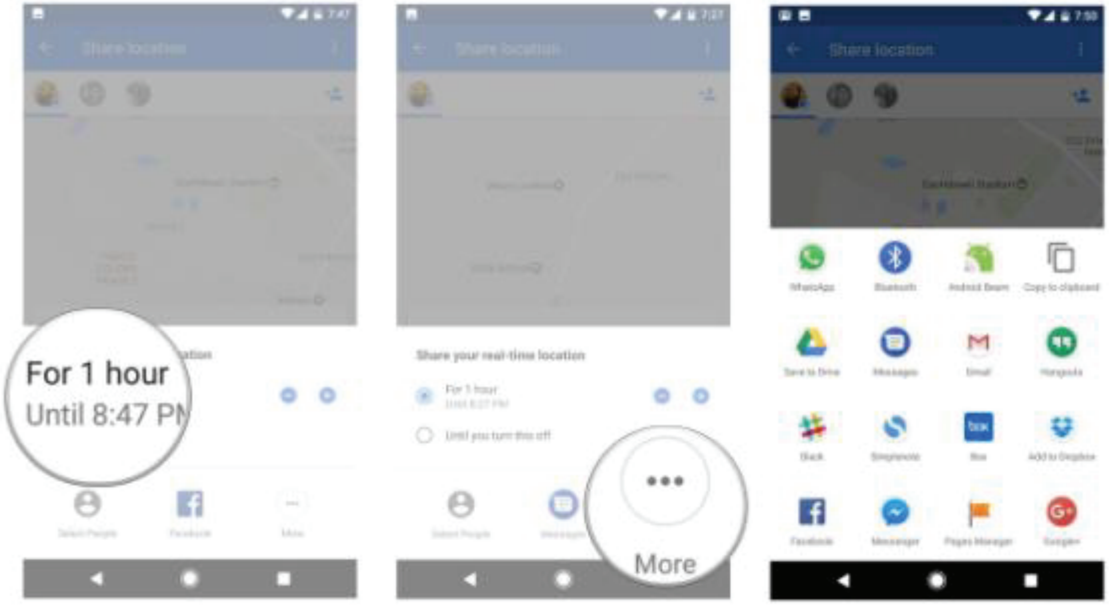
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to create a shareable link You can also create a link and use it to share your location easily. Here's how to do it: 1. Tap the **hamburger menu** on the top left corner of the screen. 2. Select **Share location**. 3. Tap **Get Started**.  Share loc Share location Let friends … where you are GET START https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	4. Select the **amount of time** you want to share your location. 5. Tap **More**. 6. Select **your app of choice** to create and send a unique URL that broadcasts your current location. You can email it, send the link via Messenger, or even tweet it to the intended recipient.  Share location For 1 hour Until 8:47 PM Share your real-time location Until you turn this off More https://www.androidcentral.com/how-share-location-google-maps

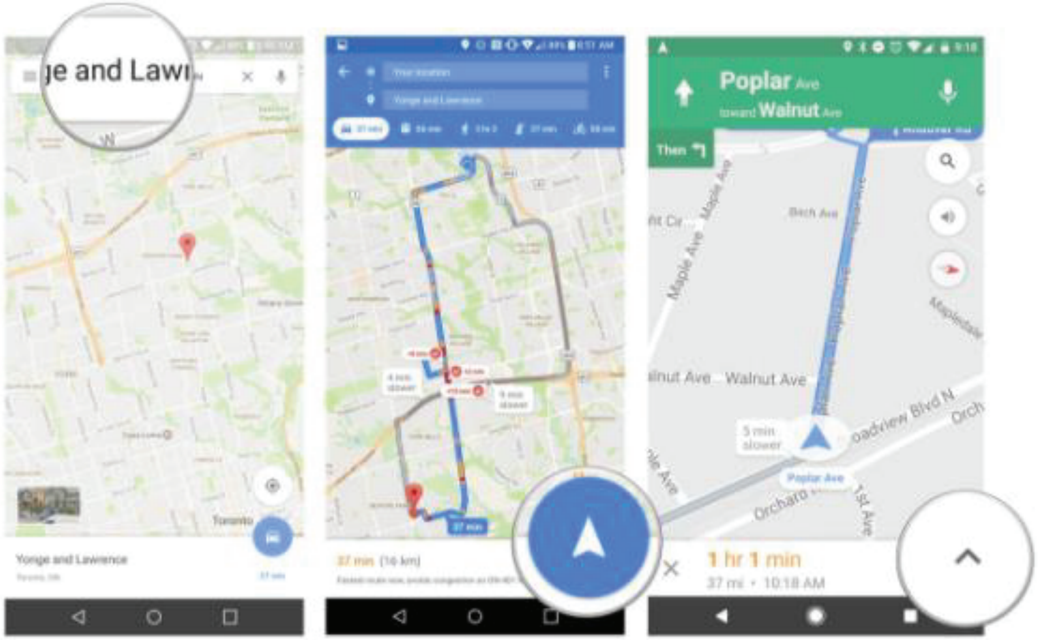
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to share your navigation directions while you walk, drive or transit One of the best ways to share your location with someone is during a drive, walk or transit. If you're meeting a friend or family member somewhere, or navigating towards their home, Google Maps lets you share your location with them for the duration of the trip. It's magic! 1. In the **search bar** enter your destination. 2. Pick your navigation type (drive, transit, walk) and press the blue **navigate button**. 3. Tap the arrow next to the **time-to-destination** number at the bottom of the screen.  https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

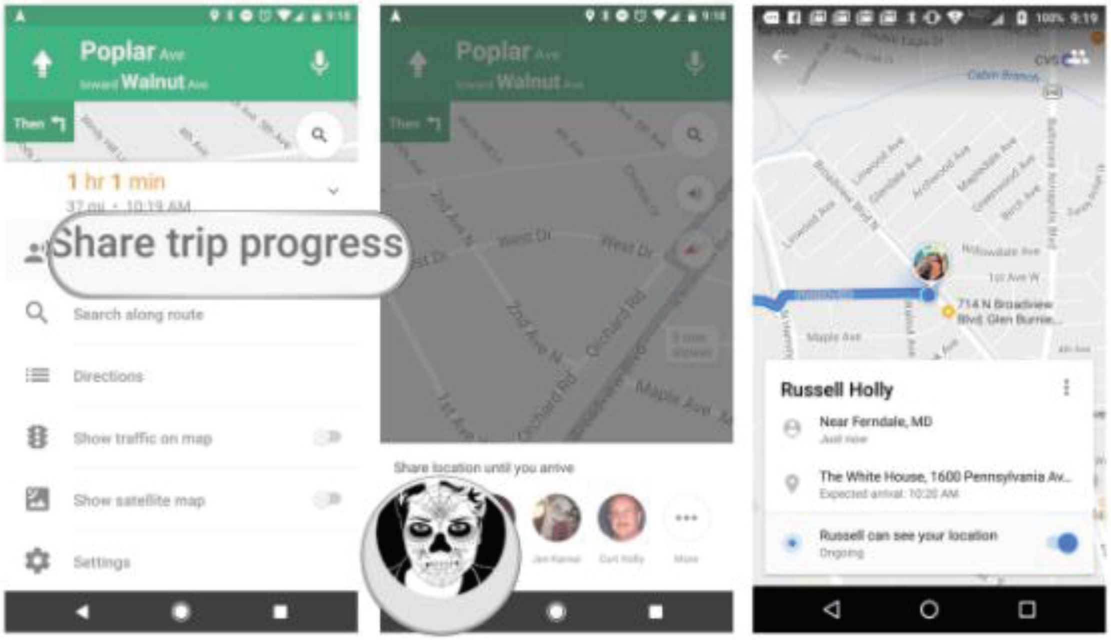
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	4. Tap Share trip progress. 5. Choose one or more contacts to share trip progress.  Poplar Ave toward Walnut Ave Then 1 hr 1 min 37 mi · 10:19 AM Share trip progress Search along route Directions Show traffic on map Show satellite map Settings Share location until you arrive More Russell Holly Near Ferndale, MD Just now The White House, 1600 Pennsylvania Av... Expected arrival: 10:20 AM Russell can see your location Ongoing You can also stop sharing your location with someone before a trip ends. https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

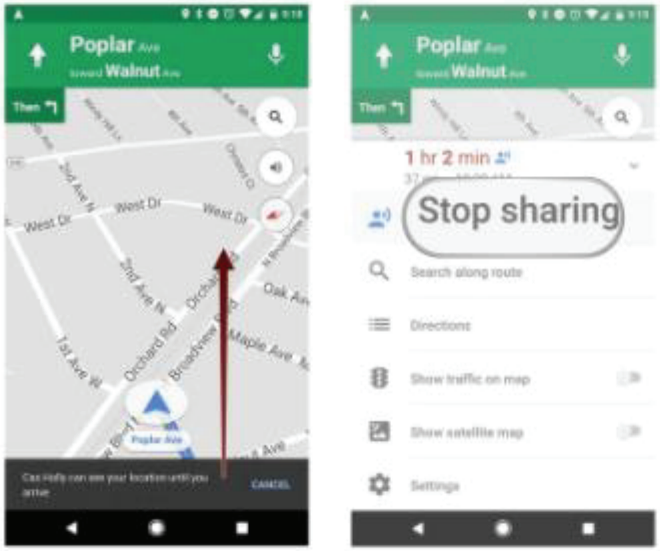
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	1. Tap the arrow next to the **time-to-destination** number at the bottom of the screen. 2. Tap **Stop sharing**.  Poplar Ave toward Walnut Ave Then West Dr 2nd Ave N 1st Ave W Orchard Rd Broadview Oak Ave Maple Ave Poplar Ave CANCEL 1 hr 2 min Stop sharing Search along route Directions Show traffic on map Show satellite map Settings That's it! Are you excited that location sharing is back in Google Maps? How often do you use the feature? https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

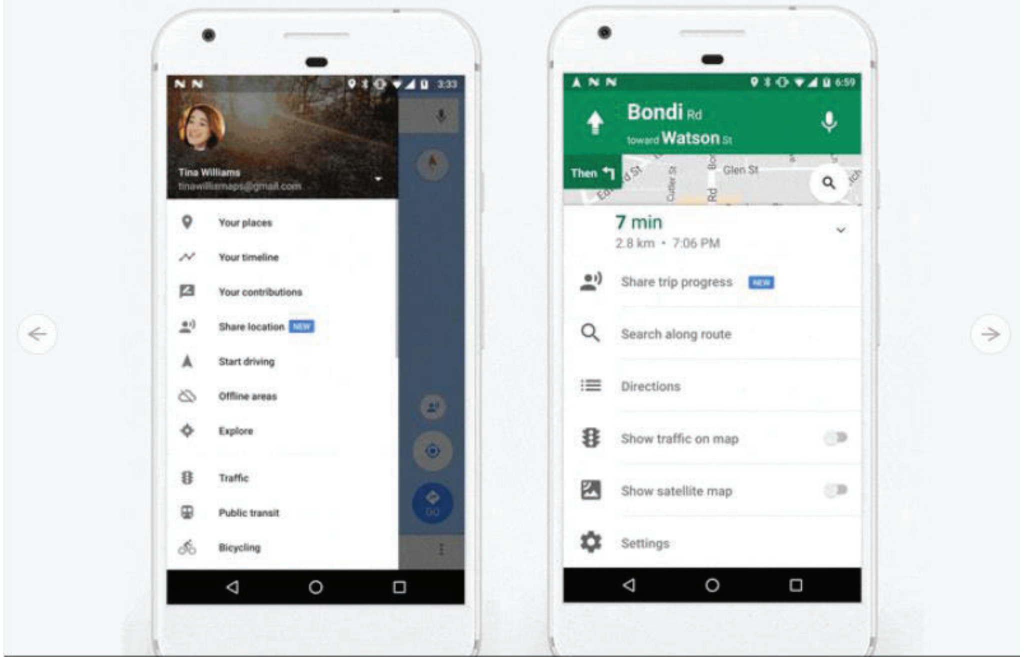
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Tina Williams tinawilliamsmaps@gmail.com Your places Your timeline Your contributions Share location NEW Start driving Offline areas Explore Traffic Public transit Bicycling Bondi Rd toward Watson St Then Glen St 7 min 2.8 km • 7:06 PM Share trip progress NEW Search along route Directions Show traffic on map Show satellite map Settings Ways to start location sharing: (left) from the app navigation drawer, and (right) from the menu, you can drag up while navigating somewhere. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

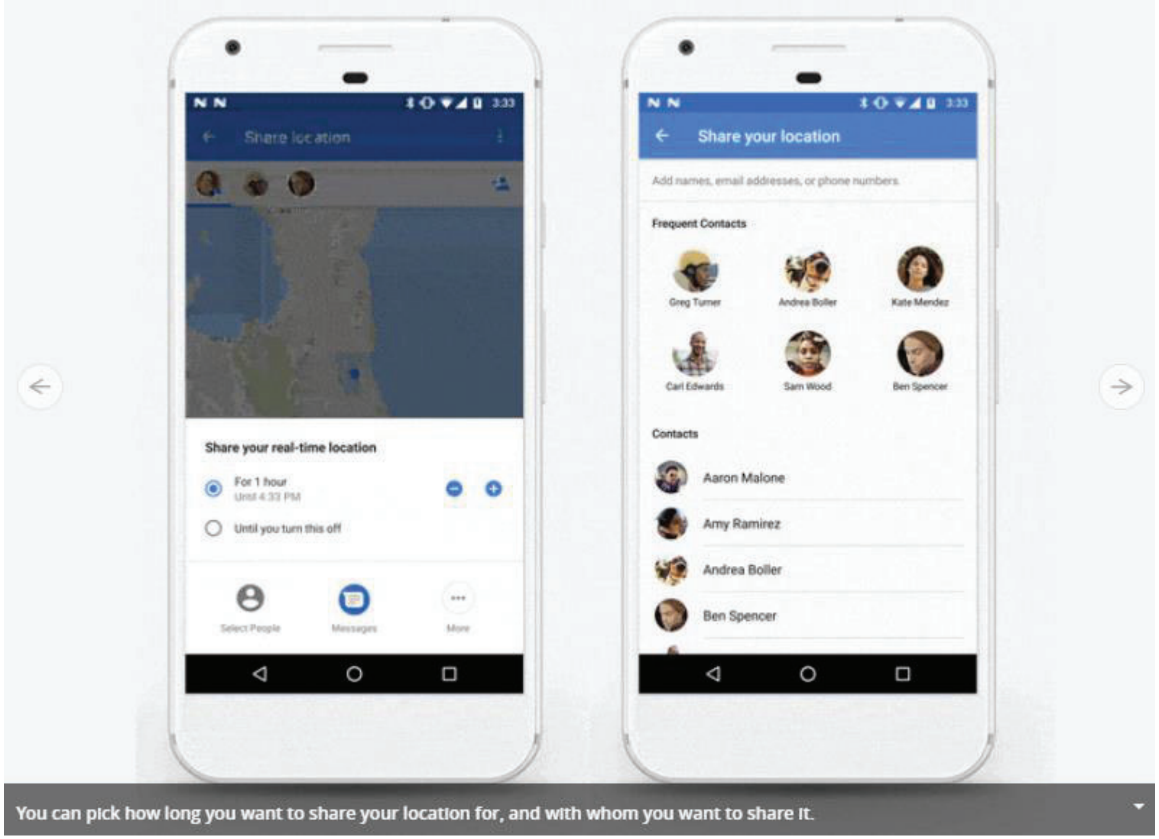
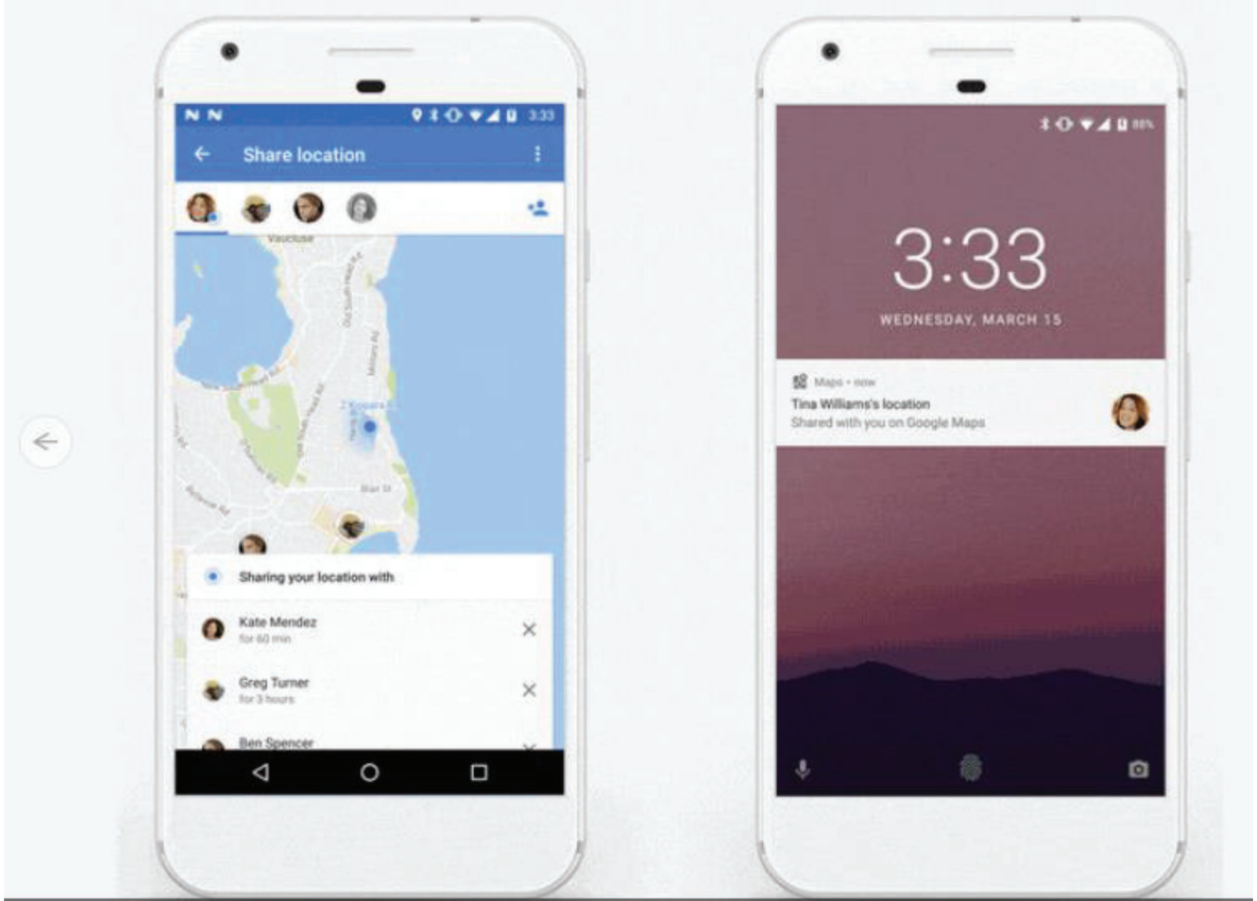
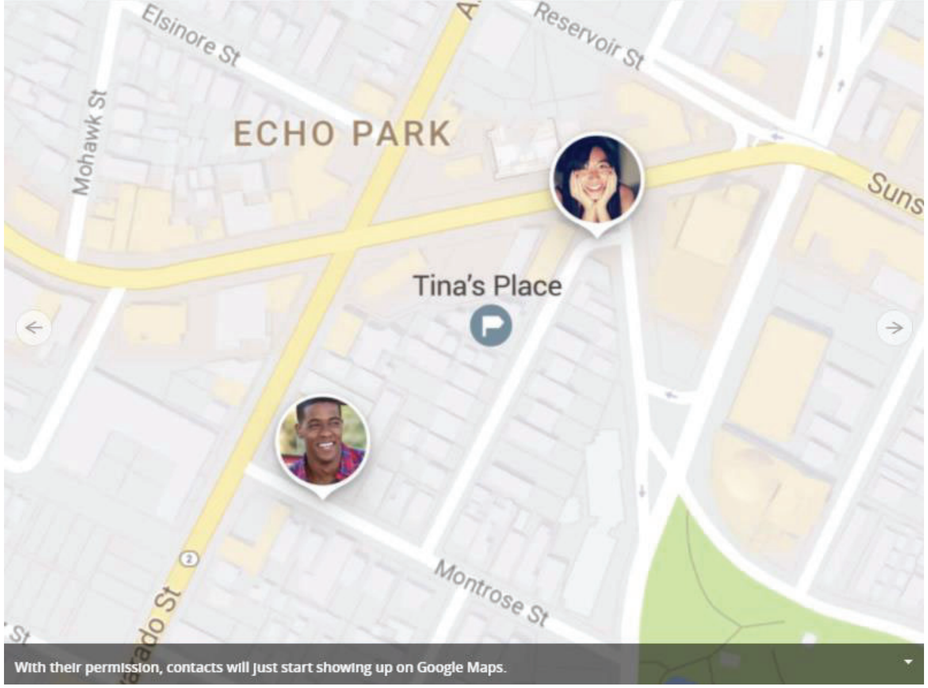
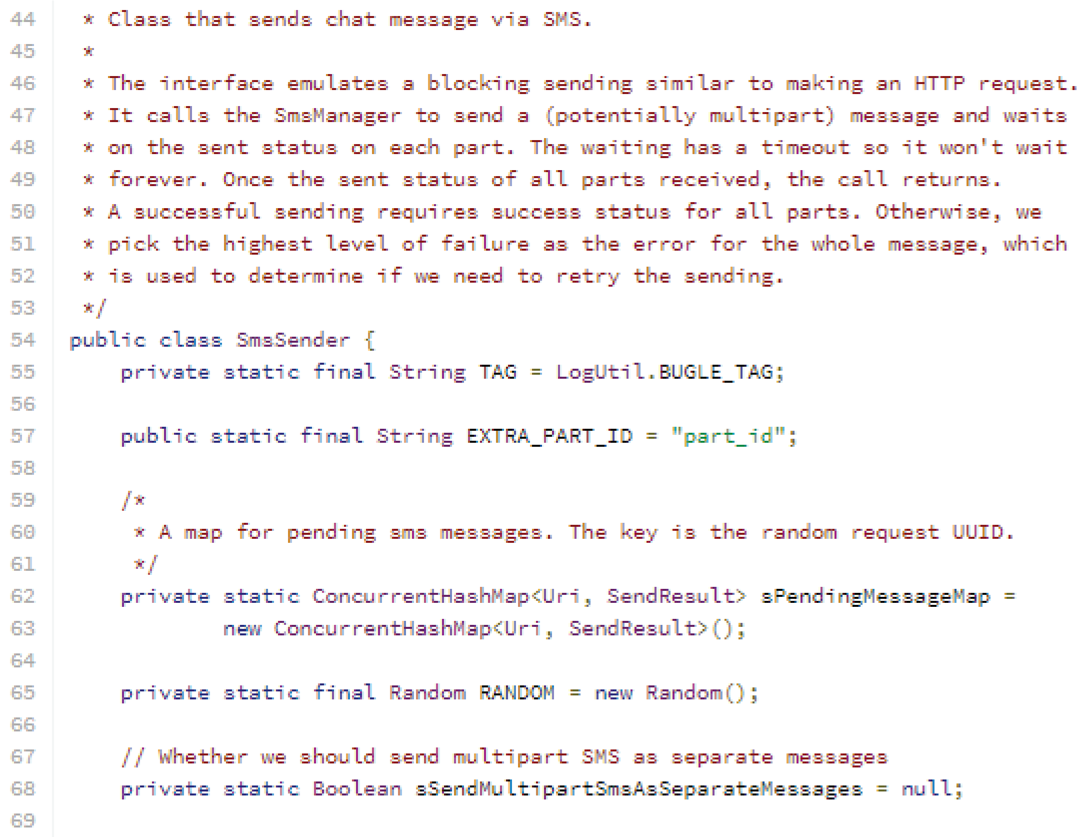
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Share location Share your real-time location For 1 hour Until 4:33 PM Until you turn this off Select People Messages More Share your location Add names, email addresses, or phone numbers Frequent Contacts Greg Turner Andrea Boller Kate Mendez Carl Edwards Sam Wood Ben Spencer Contacts Aaron Malone Amy Ramirez Andrea Boller Ben Spencer You can pick how long you want to share your location for, and with whom you want to share it. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Share location Sharing your location with Kate Mendez for 60 min Greg Turner for 3 hours Ben Spencer 3:33 WEDNESDAY, MARCH 15 Maps • now Tina Williams's location Shared with you on Google Maps People you've shared with will get a notification from Google Maps. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

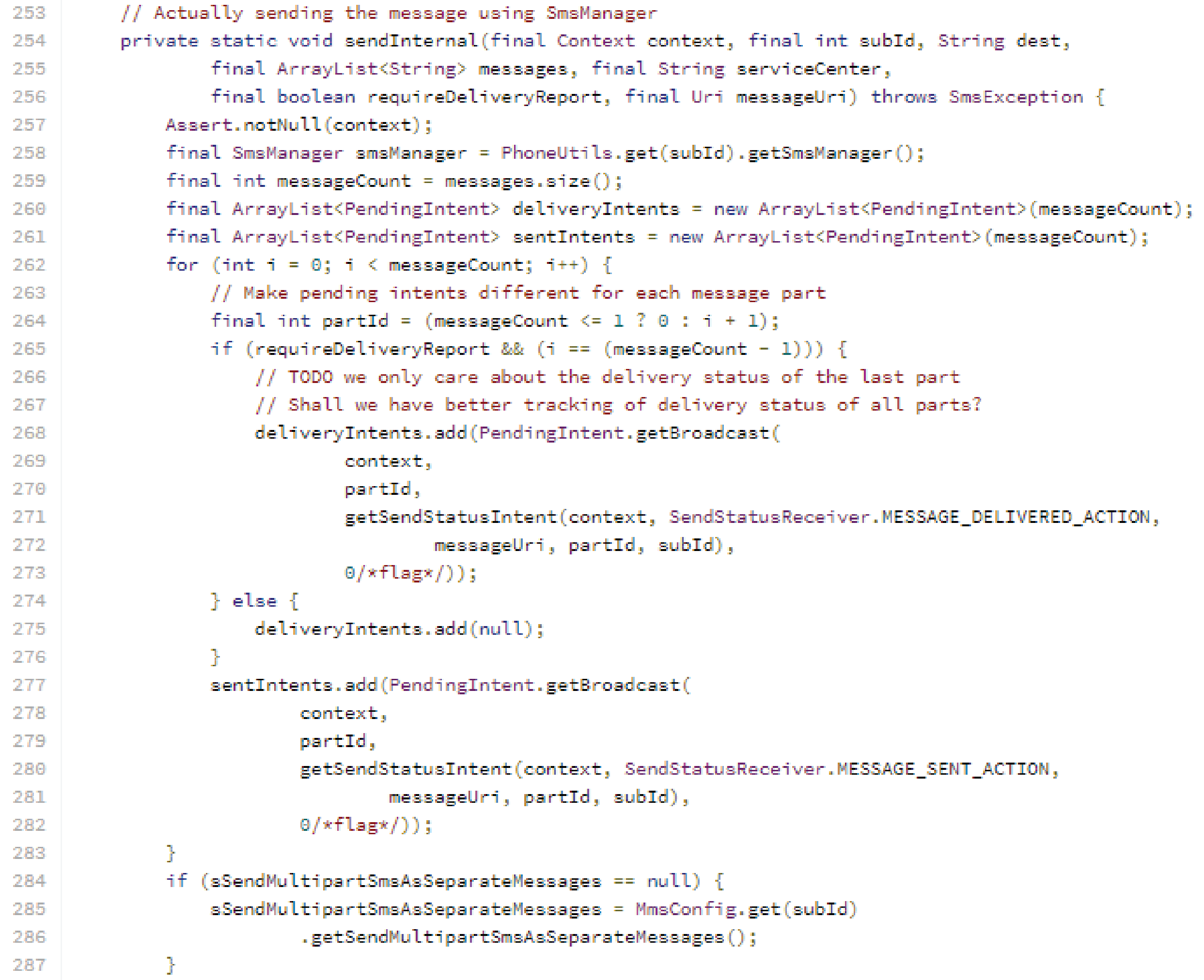
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Elsinore St Reservoir St Mohawk St ECHO PARK Suns Tina's Place Montrose St ado St With their permission, contacts will just start showing up on Google Maps. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/ **Exemplary Screenshots**: See, e.g., 1D and 1E above. **Exemplary Source Code**: The above functionality is performed at least in part by the following publicly available source code and/or source code that invokes or is invoked by the following source code (or a substantially similar copy compiled and loaded onto the Accused Products by ZTE). AGIS reserves the right to supplement these contentions with additional source code as discovery progresses and as additional source code is made available.

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>


```
 * Class that sends chat message via SMS.
 *
 * The interface emulates a blocking sending similar to making an HTTP request.
 * It calls the SmsManager to send a (potentially multipart) message and waits
 * on the sent status on each part. The waiting has a timeout so it won't wait
 * forever. Once the sent status of all parts received, the call returns.
 * A successful sending requires success status for all parts. Otherwise, we
 * pick the highest level of failure as the error for the whole message, which
 * is used to determine if we need to retry the sending.
 */
public class SmsSender {
    private static final String TAG = LogUtil.BUGLE_TAG;

    public static final String EXTRA_PART_ID = "part_id";

    /*
     * A map for pending sms messages. The key is the random request UUID.
     */
    private static ConcurrentHashMap<Uri, SendResult> sPendingMessageMap =
            new ConcurrentHashMap<Uri, SendResult>();

    private static final Random RANDOM = new Random();

    // Whether we should send multipart SMS as separate messages
    private static Boolean sSendMultipartSmsAsSeparateMessages = null;

```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/SmsSender.java
</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

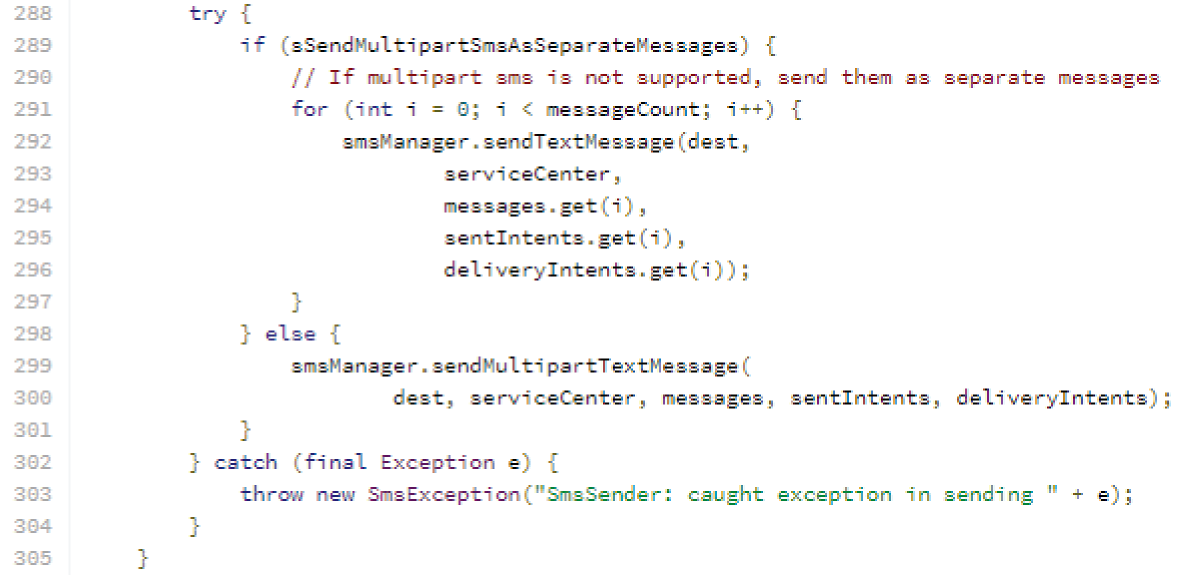
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
    // Actually sending the message using SmsManager
    private static void sendInternal(final Context context, final int subId, String dest,
            final ArrayList<String> messages, final String serviceCenter,
            final boolean requireDeliveryReport, final Uri messageUri) throws SmsException {
        Assert.notNull(context);
        final SmsManager smsManager = PhoneUtils.get(subId).getSmsManager();
        final int messageCount = messages.size();
        final ArrayList<PendingIntent> deliveryIntents = new ArrayList<PendingIntent>(messageCount);
        final ArrayList<PendingIntent> sentIntents = new ArrayList<PendingIntent>(messageCount);
        for (int i = 0; i < messageCount; i++) {
            // Make pending intents different for each message part
            final int partId = (messageCount <= 1 ? 0 : i + 1);
            if (requireDeliveryReport && (i == (messageCount - 1))) {
                // TODO we only care about the delivery status of the last part
                // Shall we have better tracking of delivery status of all parts?
                deliveryIntents.add(PendingIntent.getBroadcast(
                        context,
                        partId,
                        getSendStatusIntent(context, SendStatusReceiver.MESSAGE_DELIVERED_ACTION,
                                messageUri, partId, subId),
                        0/*flag*/));
            } else {
                deliveryIntents.add(null);
            }
            sentIntents.add(PendingIntent.getBroadcast(
                    context,
                    partId,
                    getSendStatusIntent(context, SendStatusReceiver.MESSAGE_SENT_ACTION,
                            messageUri, partId, subId),
                    0/*flag*/));
        }
        if (sSendMultipartSmsAsSeparateMessages == null) {
            sSendMultipartSmsAsSeparateMessages = MmsConfig.get(subId)
                    .getSendMultipartSmsAsSeparateMessages();
        }
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/SmsSender.java

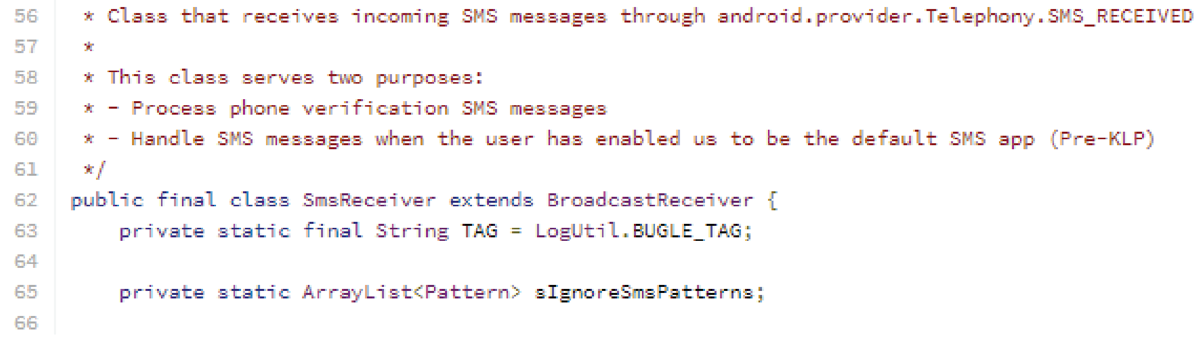
Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products



```
        try {
            if (sSendMultipartSmsAsSeparateMessages) {
                // If multipart sms is not supported, send them as separate messages
                for (int i = 0; i < messageCount; i++) {
                    smsManager.sendTextMessage(dest,
                            serviceCenter,
                            messages.get(i),
                            sentIntents.get(i),
                            deliveryIntents.get(i));
                }
            } else {
                smsManager.sendMultipartTextMessage(
                        dest, serviceCenter, messages, sentIntents, deliveryIntents);
            }
        } catch (final Exception e) {
            throw new SmsException("SmsSender: caught exception in sending " + e);
        }
    }
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/SmsSender.java

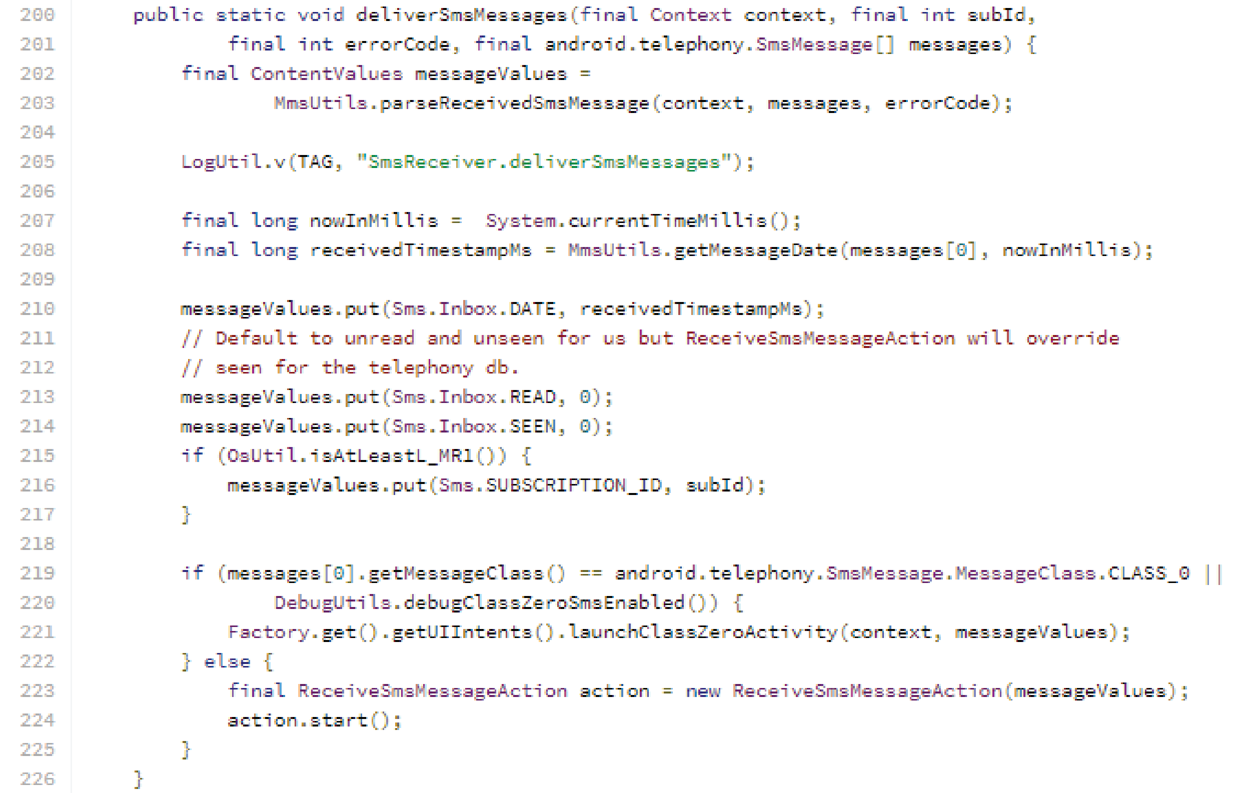


```
 * Class that receives incoming SMS messages through android.provider.Telephony.SMS_RECEIVED
 *
 * This class serves two purposes:
 * - Process phone verification SMS messages
 * - Handle SMS messages when the user has enabled us to be the default SMS app (Pre-KLP)
 */
public final class SmsReceiver extends BroadcastReceiver {
    private static final String TAG = LogUtil.BUGLE_TAG;

    private static ArrayList<Pattern> sIgnoreSmsPatterns;

```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/receiver/SmsReceiver.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>



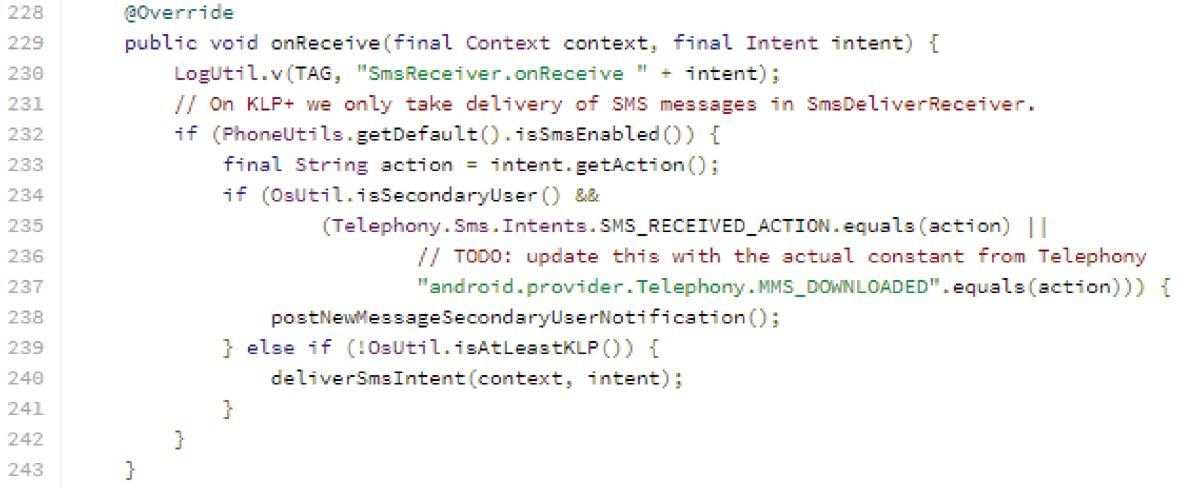
```
    public static void deliverSmsMessages(final Context context, final int subId,
            final int errorCode, final android.telephony.SmsMessage[] messages) {
        final ContentValues messageValues =
                MmsUtils.parseReceivedSmsMessage(context, messages, errorCode);

        LogUtil.v(TAG, "SmsReceiver.deliverSmsMessages");

        final long nowInMillis =  System.currentTimeMillis();
        final long receivedTimestampMs = MmsUtils.getMessageDate(messages[0], nowInMillis);

        messageValues.put(Sms.Inbox.DATE, receivedTimestampMs);
        // Default to unread and unseen for us but ReceiveSmsMessageAction will override
        // seen for the telephony db.
        messageValues.put(Sms.Inbox.READ, 0);
        messageValues.put(Sms.Inbox.SEEN, 0);
        if (OsUtil.isAtLeastL_MR1()) {
            messageValues.put(Sms.SUBSCRIPTION_ID, subId);
        }

        if (messages[0].getMessageClass() == android.telephony.SmsMessage.MessageClass.CLASS_0 ||
                DebugUtils.debugClassZeroSmsEnabled()) {
            Factory.get().getUIIntents().launchClassZeroActivity(context, messageValues);
        } else {
            final ReceiveSmsMessageAction action = new ReceiveSmsMessageAction(messageValues);
            action.start();
        }
    }
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/receiver/SmsReceiver.java

</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

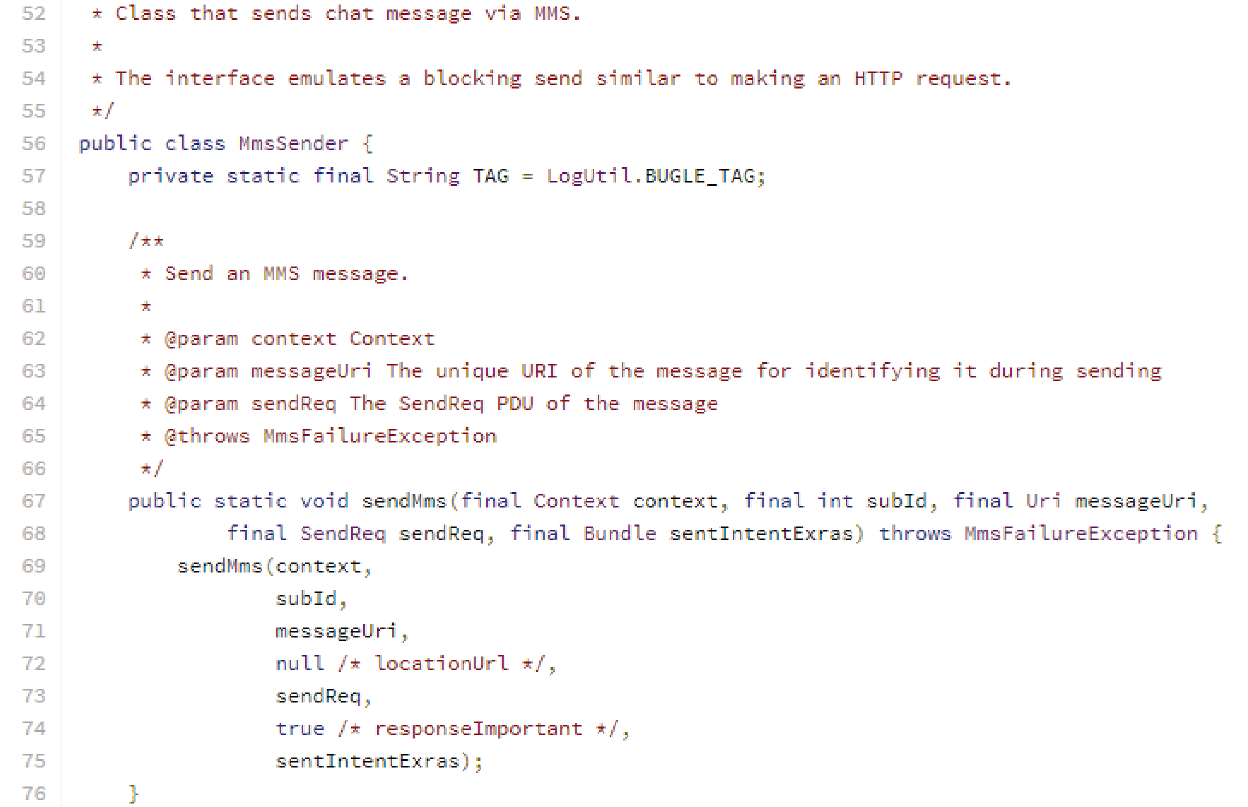
US9467838	Exemplary Supporting Evidence Regarding Accused Products



```java
    @Override
    public void onReceive(final Context context, final Intent intent) {
        LogUtil.v(TAG, "SmsReceiver.onReceive " + intent);
        // On KLP+ we only take delivery of SMS messages in SmsDeliverReceiver.
        if (PhoneUtils.getDefault().isSmsEnabled()) {
            final String action = intent.getAction();
            if (OsUtil.isSecondaryUser() &&
                    (Telephony.Sms.Intents.SMS_RECEIVED_ACTION.equals(action) ||
                            // TODO: update this with the actual constant from Telephony
                            "android.provider.Telephony.MMS_DOWNLOADED".equals(action))) {
                postNewMessageSecondaryUserNotification();
            } else if (!OsUtil.isAtLeastKLP()) {
                deliverSmsIntent(context, intent);
            }
        }
    }
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/receiver/SmsReceiver.java

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>

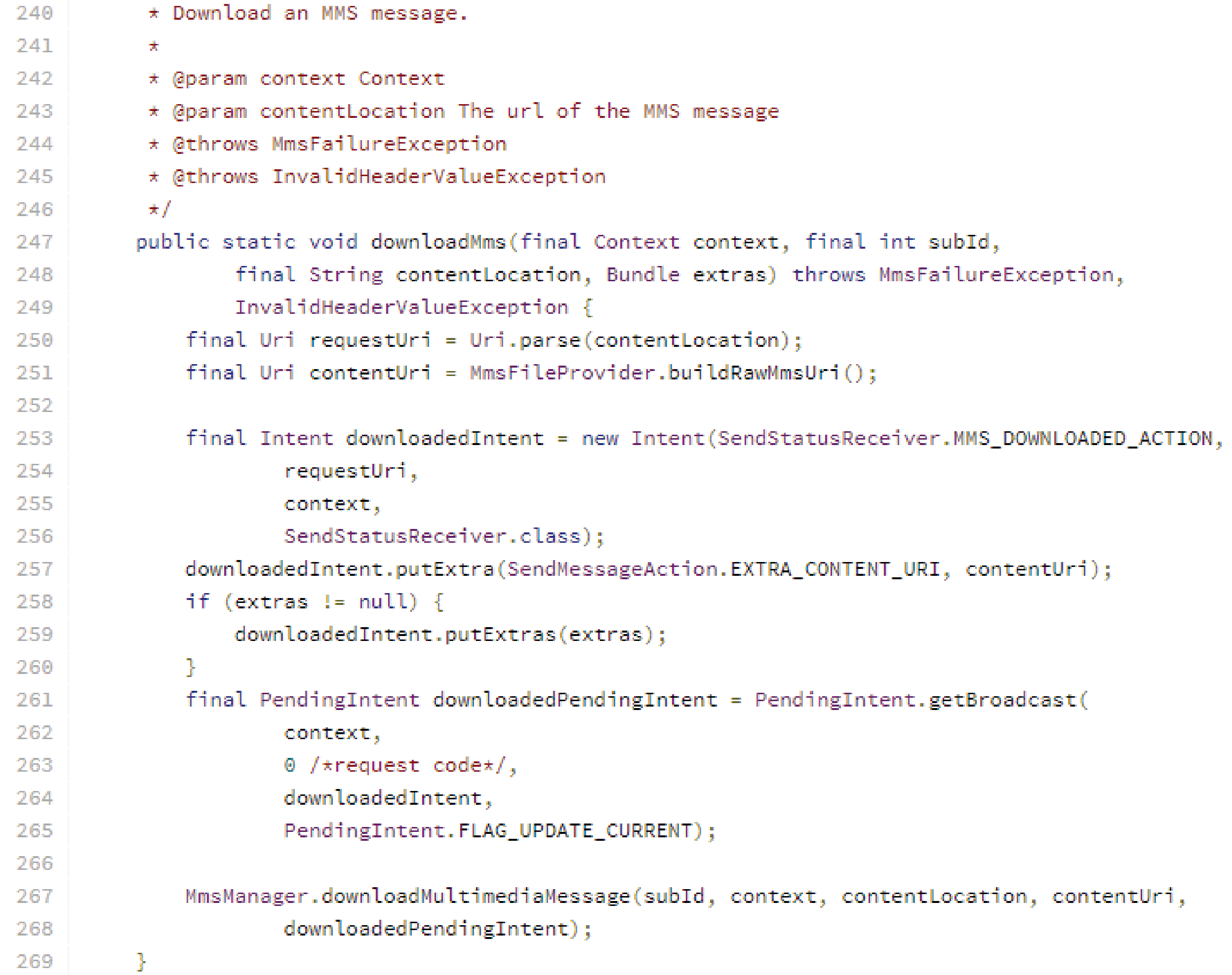


```
 * Class that sends chat message via MMS.
 *
 * The interface emulates a blocking send similar to making an HTTP request.
 */
public class MmsSender {
    private static final String TAG = LogUtil.BUGLE_TAG;

    /**
     * Send an MMS message.
     *
     * @param context Context
     * @param messageUri The unique URI of the message for identifying it during sending
     * @param sendReq The SendReq PDU of the message
     * @throws MmsFailureException
     */
    public static void sendMms(final Context context, final int subId, final Uri messageUri,
            final SendReq sendReq, final Bundle sentIntentExras) throws MmsFailureException {
        sendMms(context,
                subId,
                messageUri,
                null /* locationUrl */,
                sendReq,
                true /* responseImportant */,
                sentIntentExras);
    }
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/MmsSender.java

</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

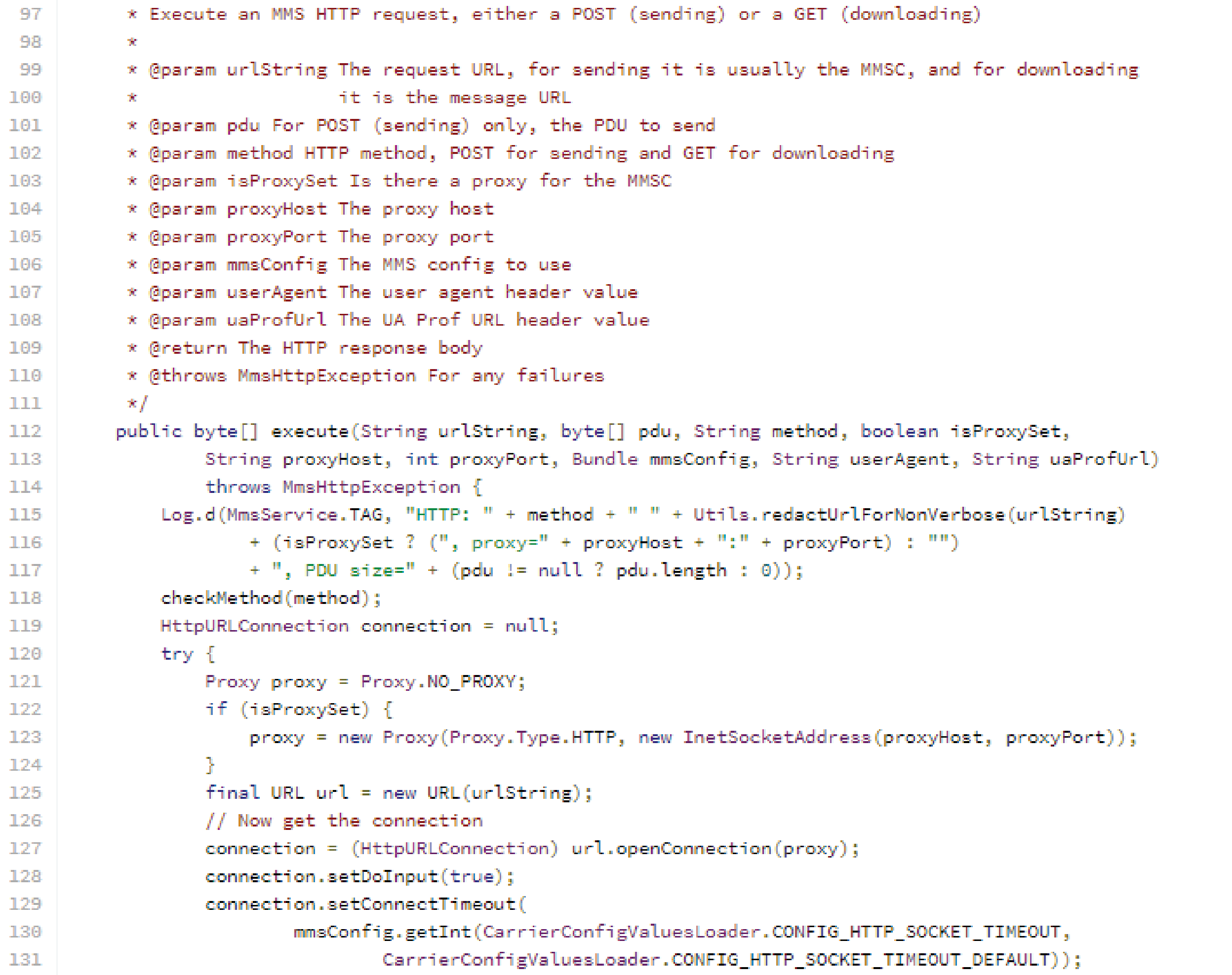
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
 * Download an MMS message.
 *
 * @param context Context
 * @param contentLocation The url of the MMS message
 * @throws MmsFailureException
 * @throws InvalidHeaderValueException
 */
public static void downloadMms(final Context context, final int subId,
        final String contentLocation, Bundle extras) throws MmsFailureException,
        InvalidHeaderValueException {
    final Uri requestUri = Uri.parse(contentLocation);
    final Uri contentUri = MmsFileProvider.buildRawMmsUri();

    final Intent downloadedIntent = new Intent(SendStatusReceiver.MMS_DOWNLOADED_ACTION,
            requestUri,
            context,
            SendStatusReceiver.class);
    downloadedIntent.putExtra(SendMessageAction.EXTRA_CONTENT_URI, contentUri);
    if (extras != null) {
        downloadedIntent.putExtras(extras);
    }
    final PendingIntent downloadedPendingIntent = PendingIntent.getBroadcast(
            context,
            0 /*request code*/,
            downloadedIntent,
            PendingIntent.FLAG_UPDATE_CURRENT);

    MmsManager.downloadMultimediaMessage(subId, context, contentLocation, contentUri,
            downloadedPendingIntent);
}
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/MmsSender.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products

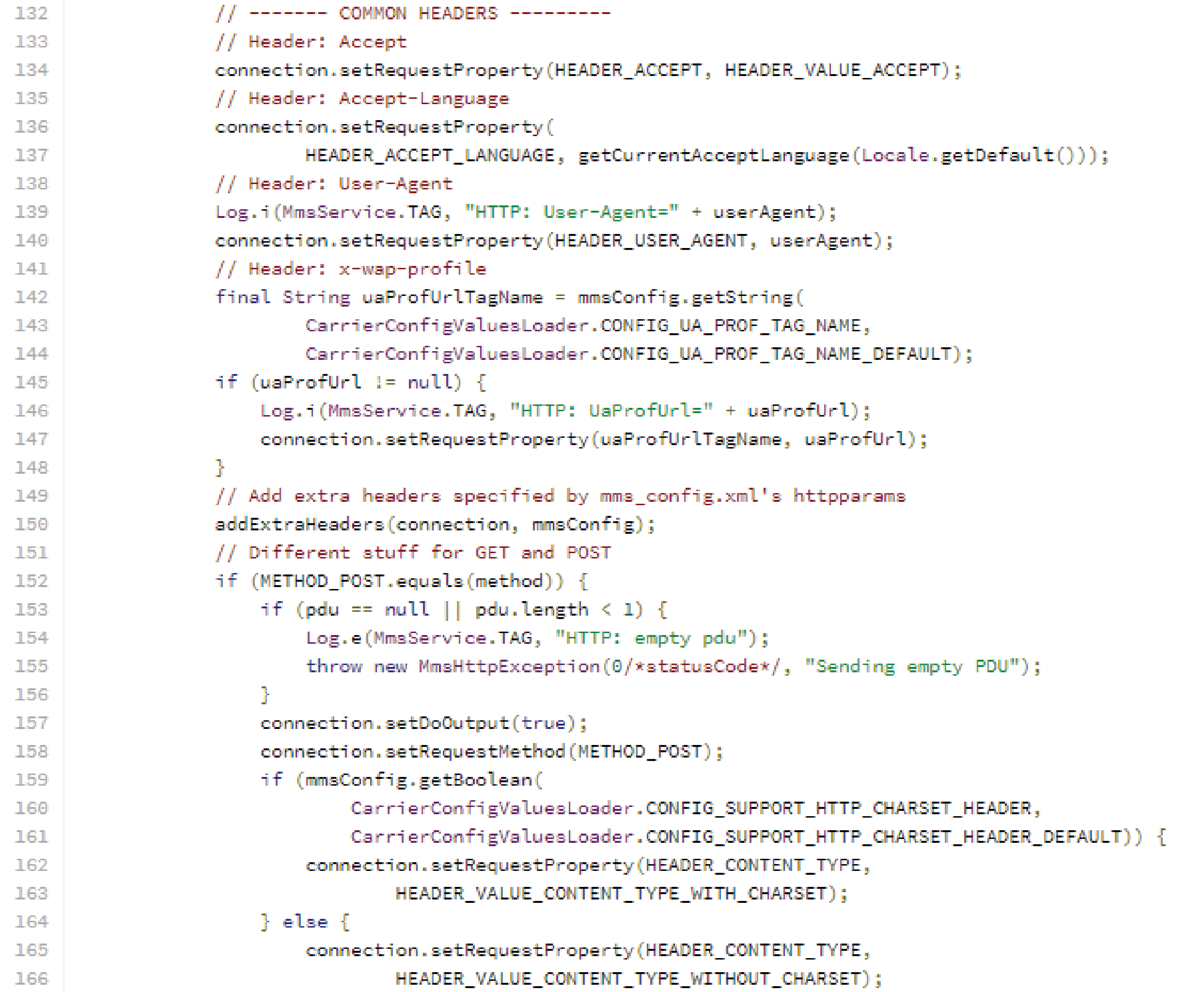


```
     * Execute an MMS HTTP request, either a POST (sending) or a GET (downloading)
     *
     * @param urlString The request URL, for sending it is usually the MMSC, and for downloading
     *                  it is the message URL
     * @param pdu For POST (sending) only, the PDU to send
     * @param method HTTP method, POST for sending and GET for downloading
     * @param isProxySet Is there a proxy for the MMSC
     * @param proxyHost The proxy host
     * @param proxyPort The proxy port
     * @param mmsConfig The MMS config to use
     * @param userAgent The user agent header value
     * @param uaProfUrl The UA Prof URL header value
     * @return The HTTP response body
     * @throws MmsHttpException For any failures
     */
    public byte[] execute(String urlString, byte[] pdu, String method, boolean isProxySet,
            String proxyHost, int proxyPort, Bundle mmsConfig, String userAgent, String uaProfUrl)
            throws MmsHttpException {
        Log.d(MmsService.TAG, "HTTP: " + method + " " + Utils.redactUrlForNonVerbose(urlString)
                + (isProxySet ? (", proxy=" + proxyHost + ":" + proxyPort) : "")
                + ", PDU size=" + (pdu != null ? pdu.length : 0));
        checkMethod(method);
        HttpURLConnection connection = null;
        try {
            Proxy proxy = Proxy.NO_PROXY;
            if (isProxySet) {
                proxy = new Proxy(Proxy.Type.HTTP, new InetSocketAddress(proxyHost, proxyPort));
            }
            final URL url = new URL(urlString);
            // Now get the connection
            connection = (HttpURLConnection) url.openConnection(proxy);
            connection.setDoInput(true);
            connection.setConnectTimeout(
                    mmsConfig.getInt(CarrierConfigValuesLoader.CONFIG_HTTP_SOCKET_TIMEOUT,
                            CarrierConfigValuesLoader.CONFIG_HTTP_SOCKET_TIMEOUT_DEFAULT));
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/android/support/v7/mms/MmsHttpClient.java

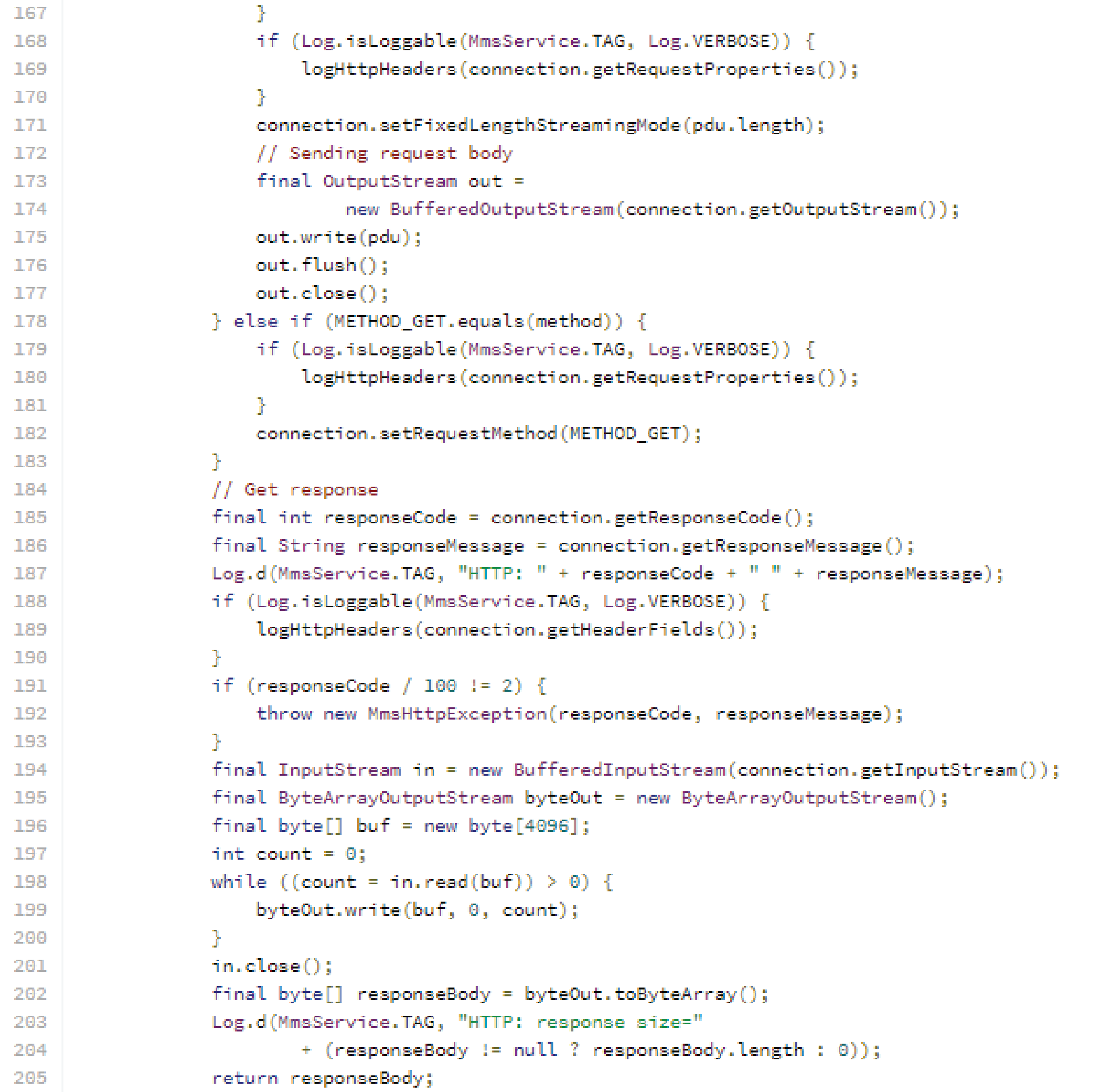
Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products



```
// ------- COMMON HEADERS ---------
// Header: Accept
connection.setRequestProperty(HEADER_ACCEPT, HEADER_VALUE_ACCEPT);
// Header: Accept-Language
connection.setRequestProperty(
        HEADER_ACCEPT_LANGUAGE, getCurrentAcceptLanguage(Locale.getDefault()));
// Header: User-Agent
Log.i(MmsService.TAG, "HTTP: User-Agent=" + userAgent);
connection.setRequestProperty(HEADER_USER_AGENT, userAgent);
// Header: x-wap-profile
final String uaProfUrlTagName = mmsConfig.getString(
        CarrierConfigValuesLoader.CONFIG_UA_PROF_TAG_NAME,
        CarrierConfigValuesLoader.CONFIG_UA_PROF_TAG_NAME_DEFAULT);
if (uaProfUrl != null) {
    Log.i(MmsService.TAG, "HTTP: UaProfUrl=" + uaProfUrl);
    connection.setRequestProperty(uaProfUrlTagName, uaProfUrl);
}
// Add extra headers specified by mms_config.xml's httpparams
addExtraHeaders(connection, mmsConfig);
// Different stuff for GET and POST
if (METHOD_POST.equals(method)) {
    if (pdu == null || pdu.length < 1) {
        Log.e(MmsService.TAG, "HTTP: empty pdu");
        throw new MmsHttpException(0/*statusCode*/, "Sending empty PDU");
    }
    connection.setDoOutput(true);
    connection.setRequestMethod(METHOD_POST);
    if (mmsConfig.getBoolean(
            CarrierConfigValuesLoader.CONFIG_SUPPORT_HTTP_CHARSET_HEADER,
            CarrierConfigValuesLoader.CONFIG_SUPPORT_HTTP_CHARSET_HEADER_DEFAULT)) {
        connection.setRequestProperty(HEADER_CONTENT_TYPE,
                HEADER_VALUE_CONTENT_TYPE_WITH_CHARSET);
    } else {
        connection.setRequestProperty(HEADER_CONTENT_TYPE,
                HEADER_VALUE_CONTENT_TYPE_WITHOUT_CHARSET);
```

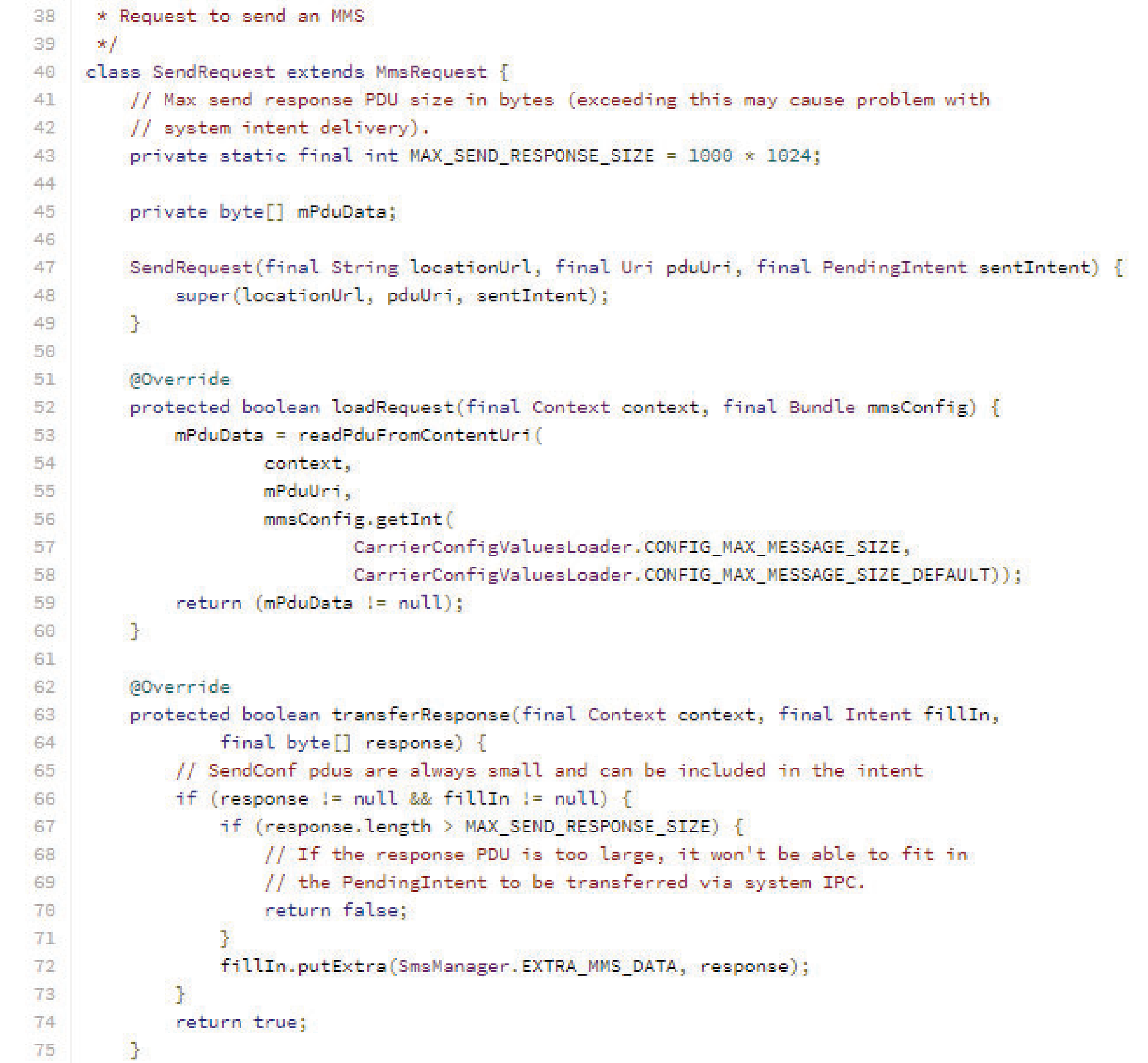
https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/android/support/v7/mms/MmsHttpClient.java

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
            }
            if (Log.isLoggable(MmsService.TAG, Log.VERBOSE)) {
                logHttpHeaders(connection.getRequestProperties());
            }
            connection.setFixedLengthStreamingMode(pdu.length);
            // Sending request body
            final OutputStream out =
                    new BufferedOutputStream(connection.getOutputStream());
            out.write(pdu);
            out.flush();
            out.close();
        } else if (METHOD_GET.equals(method)) {
            if (Log.isLoggable(MmsService.TAG, Log.VERBOSE)) {
                logHttpHeaders(connection.getRequestProperties());
            }
            connection.setRequestMethod(METHOD_GET);
        }
        // Get response
        final int responseCode = connection.getResponseCode();
        final String responseMessage = connection.getResponseMessage();
        Log.d(MmsService.TAG, "HTTP: " + responseCode + " " + responseMessage);
        if (Log.isLoggable(MmsService.TAG, Log.VERBOSE)) {
            logHttpHeaders(connection.getHeaderFields());
        }
        if (responseCode / 100 != 2) {
            throw new MmsHttpException(responseCode, responseMessage);
        }
        final InputStream in = new BufferedInputStream(connection.getInputStream());
        final ByteArrayOutputStream byteOut = new ByteArrayOutputStream();
        final byte[] buf = new byte[4096];
        int count = 0;
        while ((count = in.read(buf)) > 0) {
            byteOut.write(buf, 0, count);
        }
        in.close();
        final byte[] responseBody = byteOut.toByteArray();
        Log.d(MmsService.TAG, "HTTP: response size="
                + (responseBody != null ? responseBody.length : 0));
        return responseBody;
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/android/support/v7/mms/MmsHttpClient.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
 * Request to send an MMS
 */
class SendRequest extends MmsRequest {
    // Max send response PDU size in bytes (exceeding this may cause problem with
    // system intent delivery).
    private static final int MAX_SEND_RESPONSE_SIZE = 1000 * 1024;

    private byte[] mPduData;

    SendRequest(final String locationUrl, final Uri pduUri, final PendingIntent sentIntent) {
        super(locationUrl, pduUri, sentIntent);
    }

    @Override
    protected boolean loadRequest(final Context context, final Bundle mmsConfig) {
        mPduData = readPduFromContentUri(
                context,
                mPduUri,
                mmsConfig.getInt(
                        CarrierConfigValuesLoader.CONFIG_MAX_MESSAGE_SIZE,
                        CarrierConfigValuesLoader.CONFIG_MAX_MESSAGE_SIZE_DEFAULT));
        return (mPduData != null);
    }

    @Override
    protected boolean transferResponse(final Context context, final Intent fillIn,
            final byte[] response) {
        // SendConf pdus are always small and can be included in the intent
        if (response != null && fillIn != null) {
            if (response.length > MAX_SEND_RESPONSE_SIZE) {
                // If the response PDU is too large, it won't be able to fit in
                // the PendingIntent to be transferred via system IPC.
                return false;
            }
            fillIn.putExtra(SmsManager.EXTRA_MMS_DATA, response);
        }
        return true;
    }
```

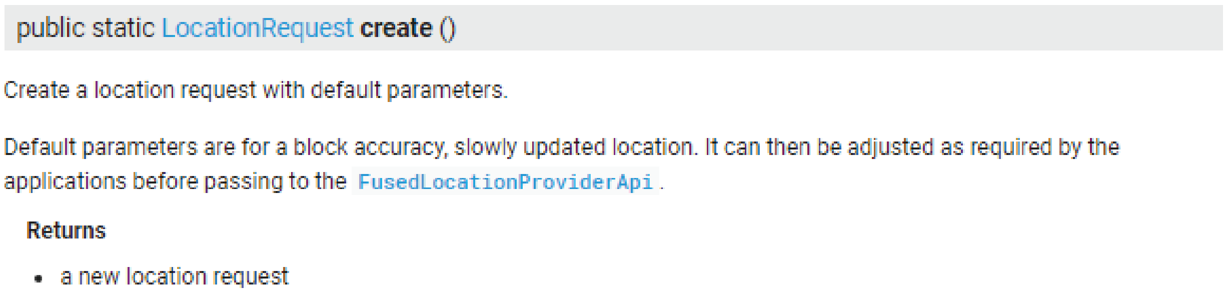
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/android/support/v7/mms/SendRequest.java  public static LocationRequest **create** () Create a location request with default parameters. Default parameters are for a block accuracy, slowly updated location. It can then be adjusted as required by the applications before passing to the `FusedLocationProviderApi`. **Returns** - a new location request https://developers.google.com/android/reference/com/google/android/gms/location/LocationRequest

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

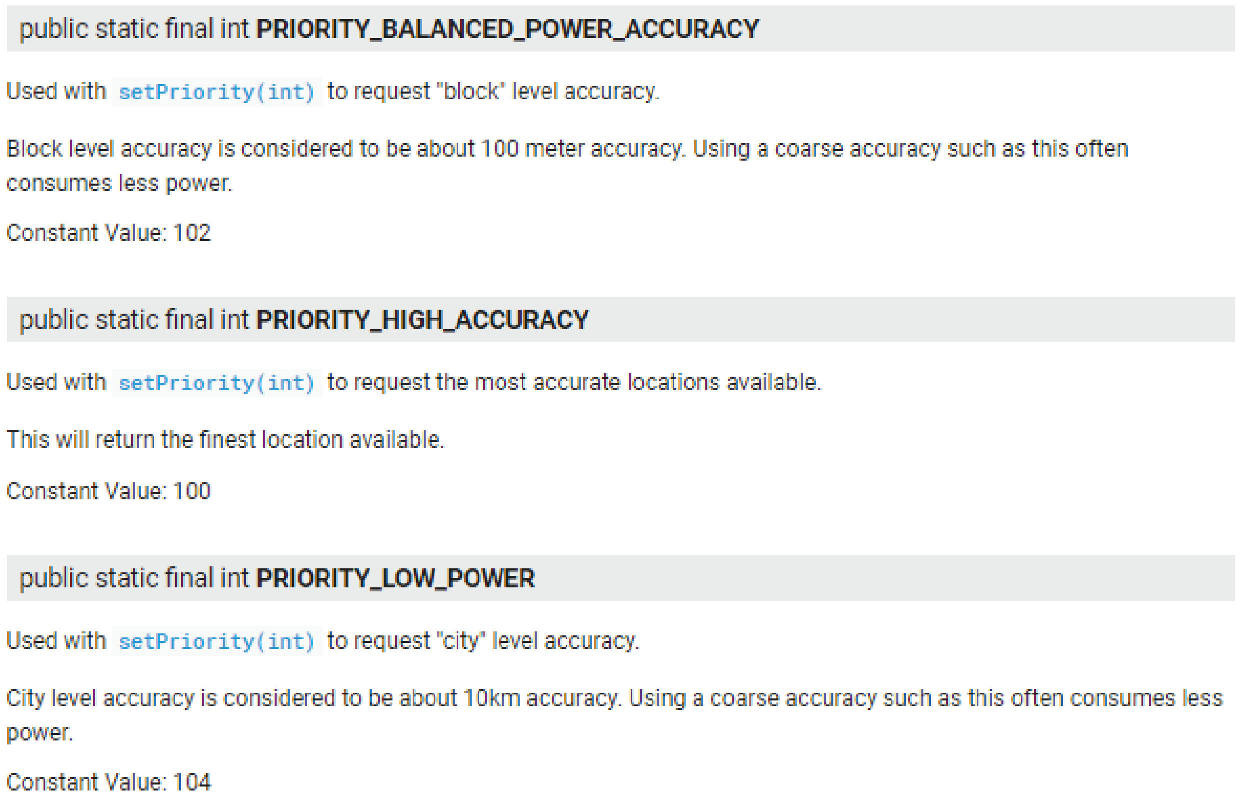
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public static final int **PRIORITY_BALANCED_POWER_ACCURACY** Used with `setPriority(int)` to request "block" level accuracy. Block level accuracy is considered to be about 100 meter accuracy. Using a coarse accuracy such as this often consumes less power. Constant Value: 102 public static final int **PRIORITY_HIGH_ACCURACY** Used with `setPriority(int)` to request the most accurate locations available. This will return the finest location available. Constant Value: 100 public static final int **PRIORITY_LOW_POWER** Used with `setPriority(int)` to request "city" level accuracy. City level accuracy is considered to be about 10km accuracy. Using a coarse accuracy such as this often consumes less power. Constant Value: 104 https://developers.google.com/android/reference/com/google/android/gms/location/LocationRequest

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

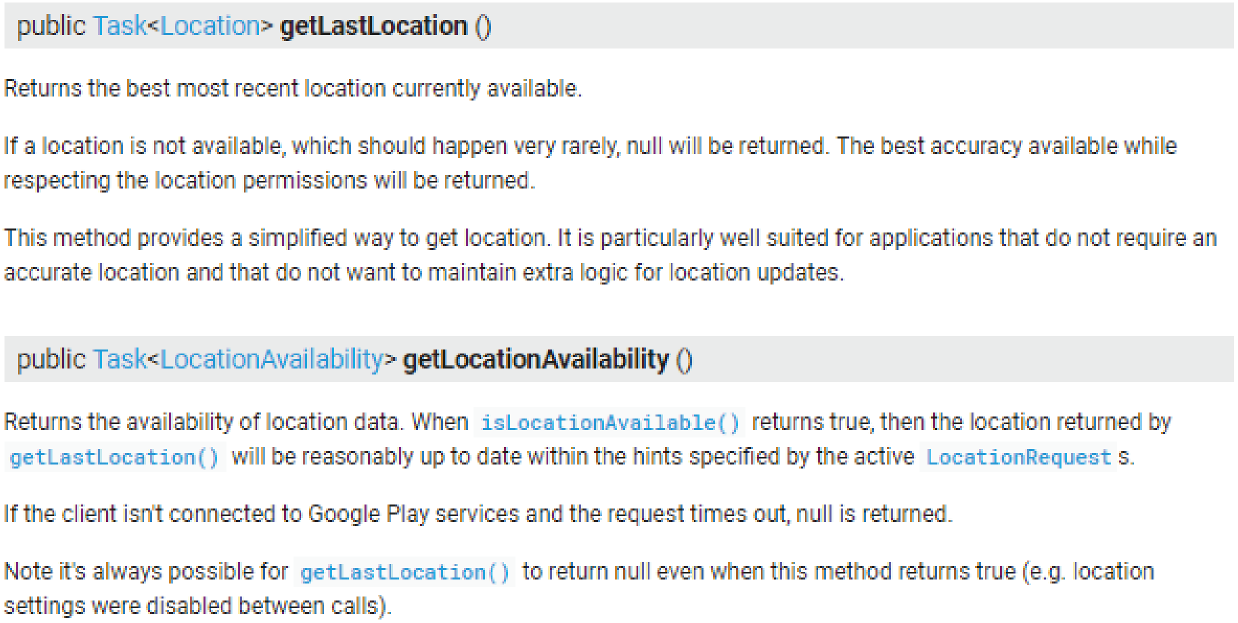
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public Task<Location> **getLastLocation** () Returns the best most recent location currently available. If a location is not available, which should happen very rarely, null will be returned. The best accuracy available while respecting the location permissions will be returned. This method provides a simplified way to get location. It is particularly well suited for applications that do not require an accurate location and that do not want to maintain extra logic for location updates. public Task<LocationAvailability> **getLocationAvailability** () Returns the availability of location data. When `isLocationAvailable()` returns true, then the location returned by `getLastLocation()` will be reasonably up to date within the hints specified by the active `LocationRequest` s. If the client isn't connected to Google Play services and the request times out, null is returned. Note it's always possible for `getLastLocation()` to return null even when this method returns true (e.g. location settings were disabled between calls). https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

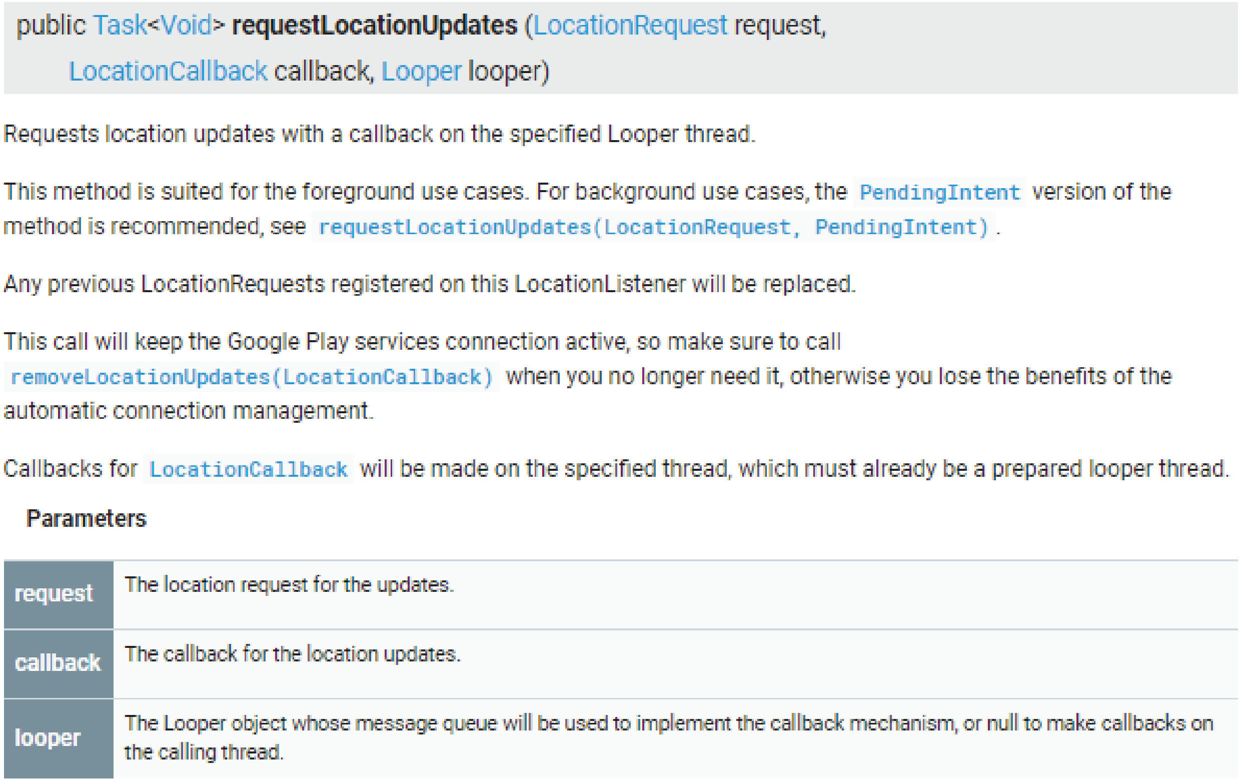
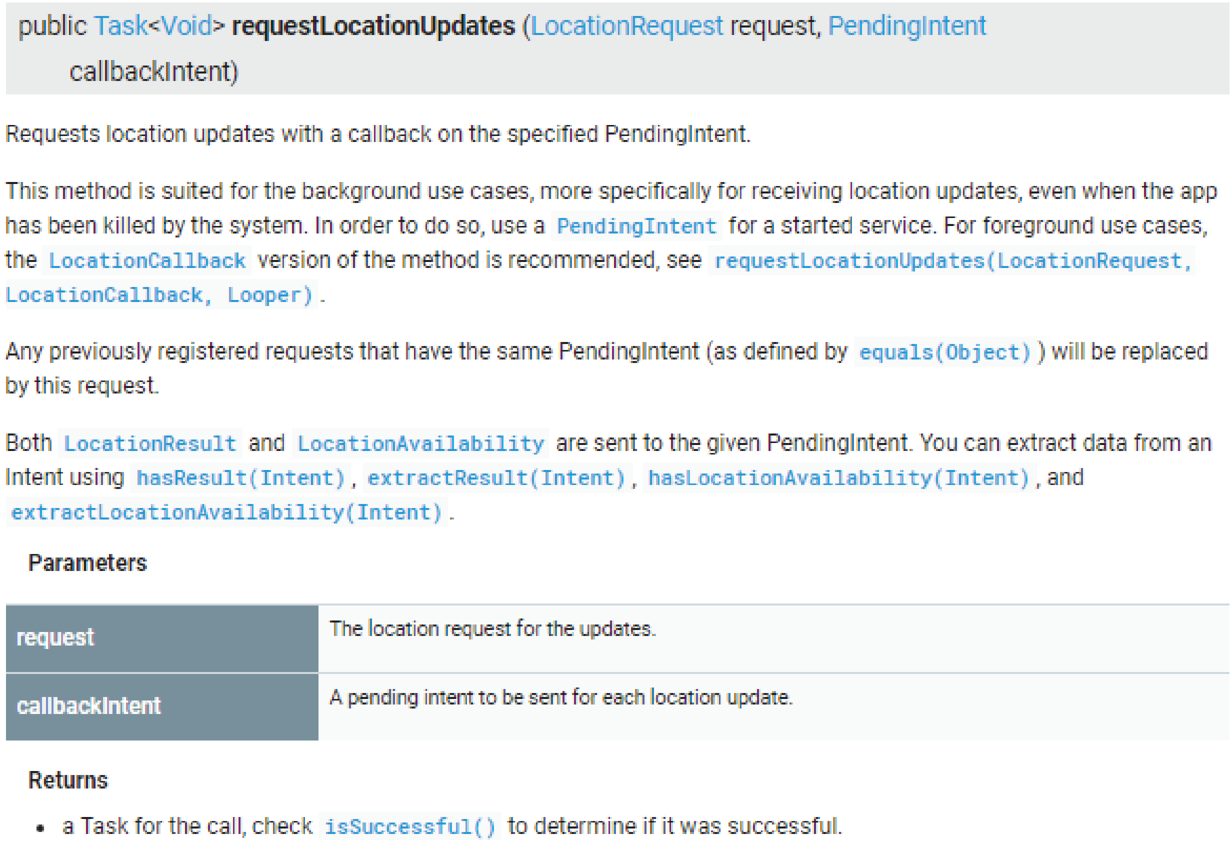
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public Task<Void> **requestLocationUpdates** (LocationRequest request, LocationCallback callback, Looper looper) Requests location updates with a callback on the specified Looper thread. This method is suited for the foreground use cases. For background use cases, the `PendingIntent` version of the method is recommended, see `requestLocationUpdates(LocationRequest, PendingIntent)` . Any previous LocationRequests registered on this LocationListener will be replaced. This call will keep the Google Play services connection active, so make sure to call `removeLocationUpdates(LocationCallback)` when you no longer need it, otherwise you lose the benefits of the automatic connection management. Callbacks for `LocationCallback` will be made on the specified thread, which must already be a prepared looper thread. **Parameters** **request** The location request for the updates. **callback** The callback for the location updates. **looper** The Looper object whose message queue will be used to implement the callback mechanism, or null to make callbacks on the calling thread. https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>


public Task<Void> **requestLocationUpdates** (LocationRequest request, PendingIntent callbackIntent)

Requests location updates with a callback on the specified PendingIntent.

This method is suited for the background use cases, more specifically for receiving location updates, even when the app has been killed by the system. In order to do so, use a `PendingIntent` for a started service. For foreground use cases, the `LocationCallback` version of the method is recommended, see `requestLocationUpdates(LocationRequest, LocationCallback, Looper)`.

Any previously registered requests that have the same PendingIntent (as defined by `equals(Object)`) will be replaced by this request.

Both `LocationResult` and `LocationAvailability` are sent to the given PendingIntent. You can extract data from an Intent using `hasResult(Intent)`, `extractResult(Intent)`, `hasLocationAvailability(Intent)`, and `extractLocationAvailability(Intent)`.

Parameters

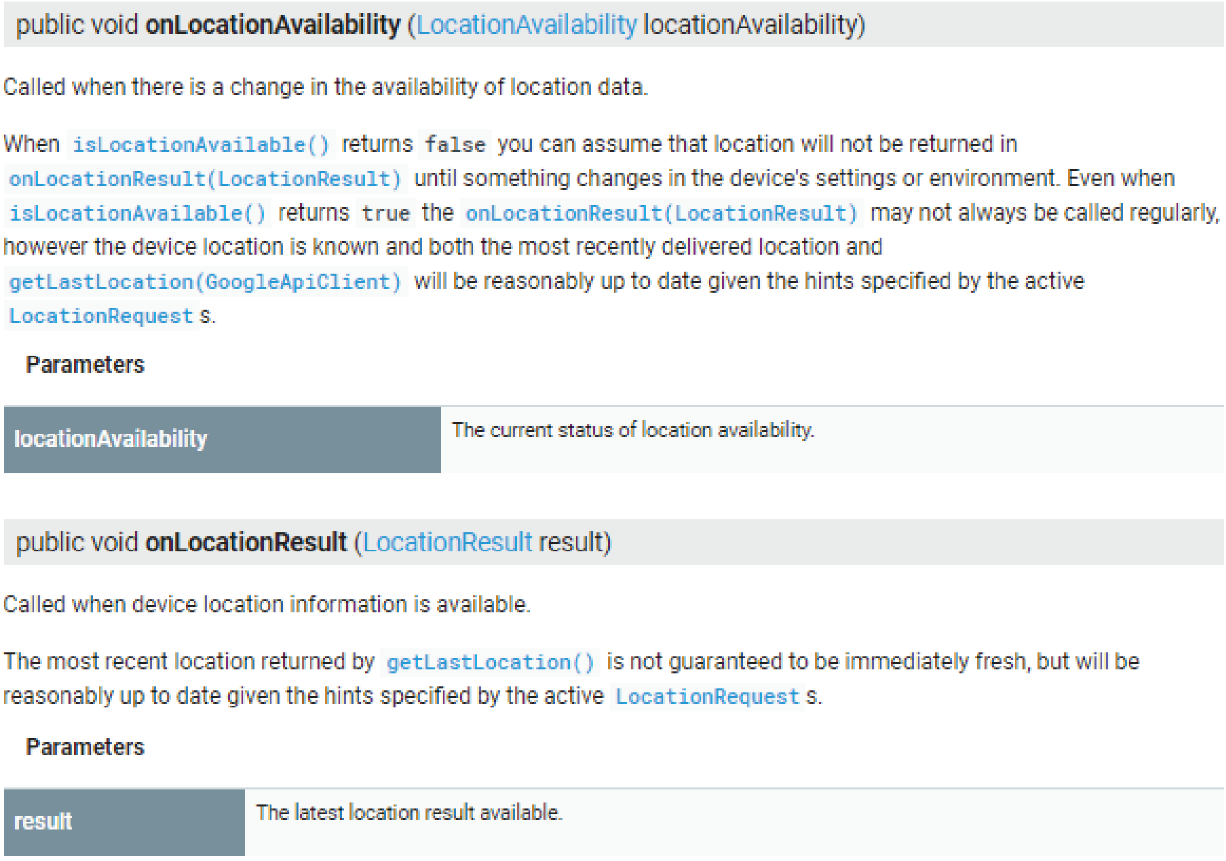
<table>
<tr><td><b>request</b></td><td>The location request for the updates.</td></tr>
<tr><td><b>callbackIntent</b></td><td>A pending intent to be sent for each location update.</td></tr>
</table>

Returns

- a Task for the call, check `isSuccessful()` to determine if it was successful.

https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient
</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>


public void **onLocationAvailability** (LocationAvailability locationAvailability)

Called when there is a change in the availability of location data.

When `isLocationAvailable()` returns `false` you can assume that location will not be returned in `onLocationResult(LocationResult)` until something changes in the device's settings or environment. Even when `isLocationAvailable()` returns `true` the `onLocationResult(LocationResult)` may not always be called regularly, however the device location is known and both the most recently delivered location and `getLastLocation(GoogleApiClient)` will be reasonably up to date given the hints specified by the active `LocationRequest` s.

Parameters

locationAvailability	The current status of location availability.

public void **onLocationResult** (LocationResult result)

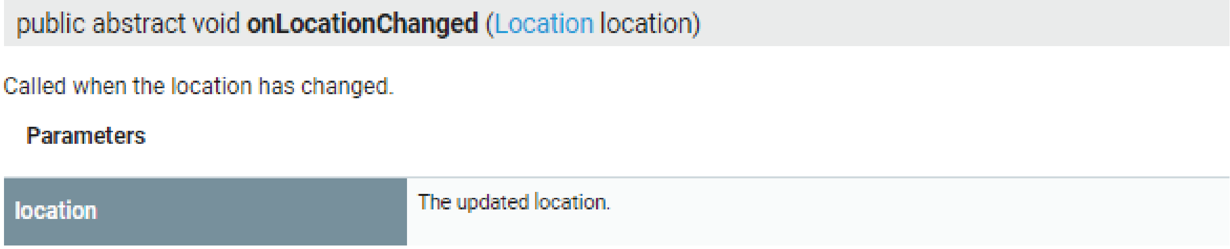
Called when device location information is available.

The most recent location returned by `getLastLocation()` is not guaranteed to be immediately fresh, but will be reasonably up to date given the hints specified by the active `LocationRequest` s.

Parameters

result	The latest location result available.

https://developers.google.com/android/reference/com/google/android/gms/location/LocationCallback



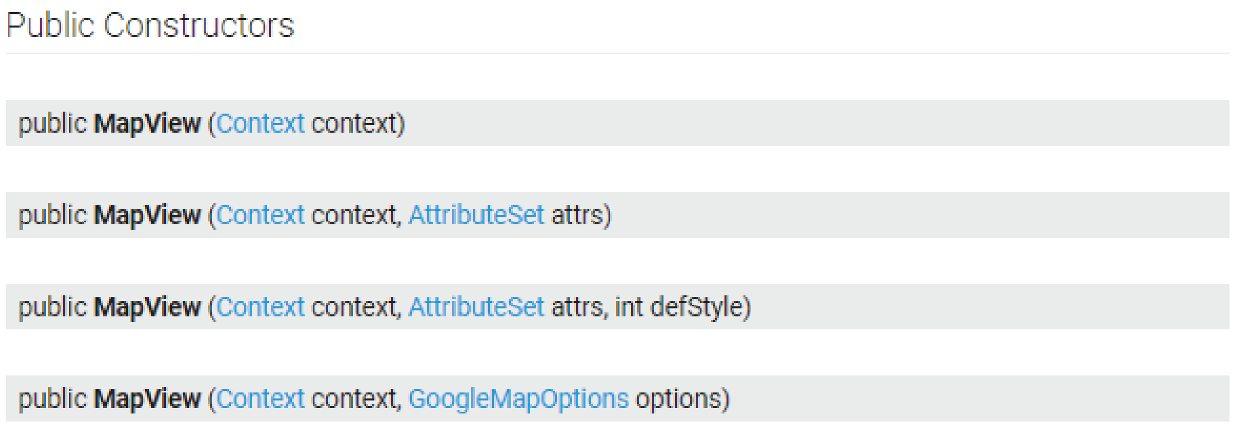
public abstract void **onLocationChanged** (Location location)

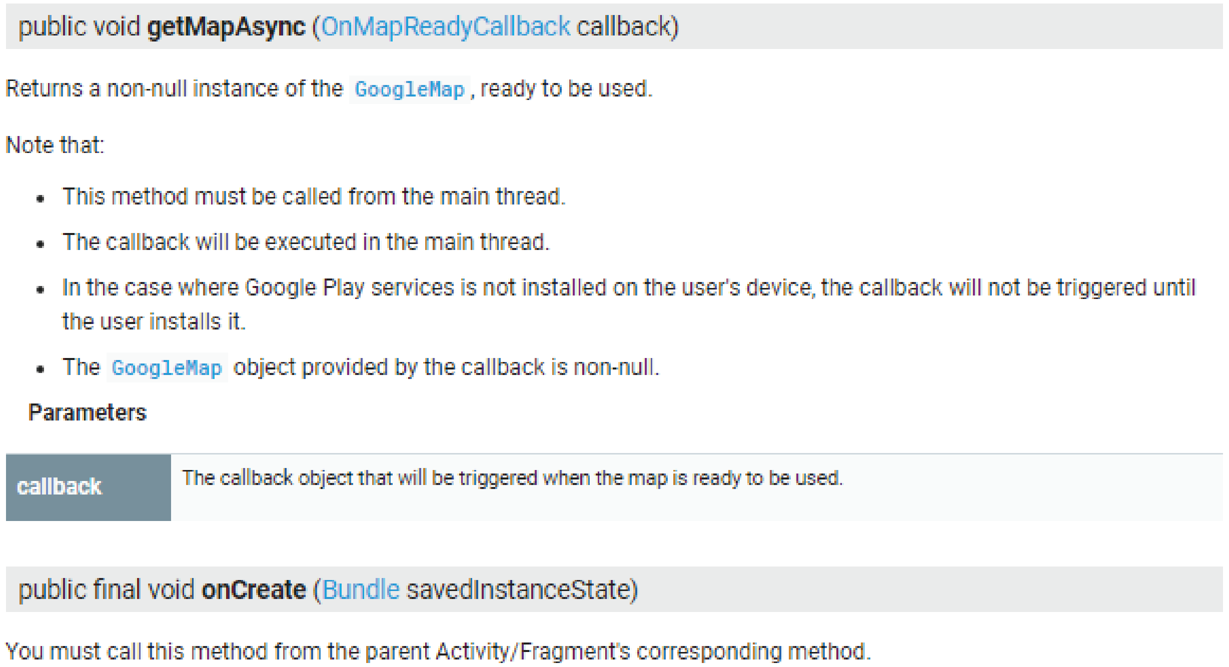
Called when the location has changed.

Parameters

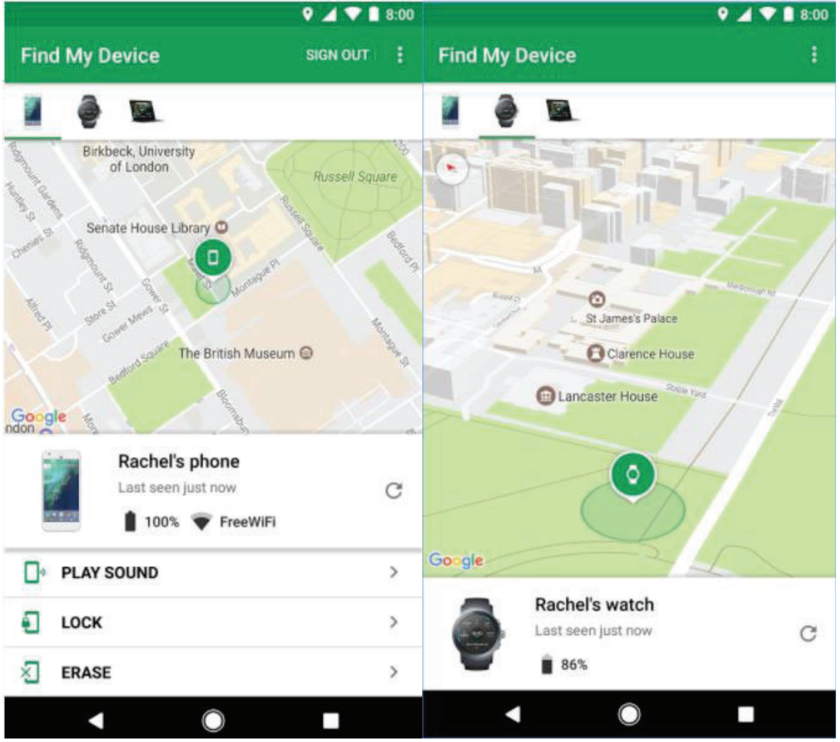
location	The updated location.
</td></tr>
</table>

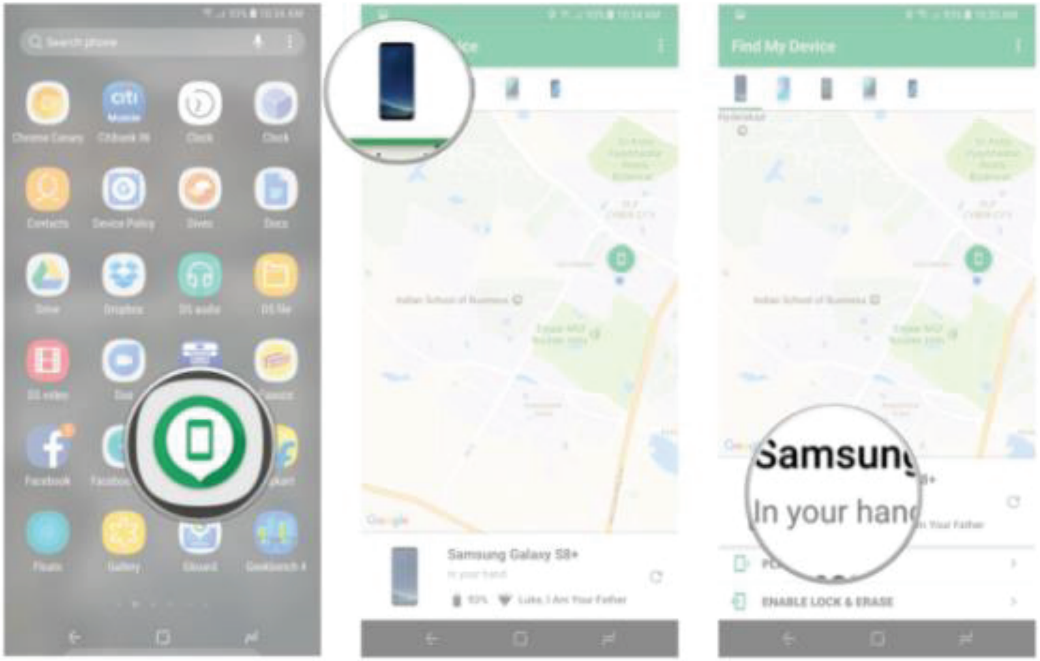
Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

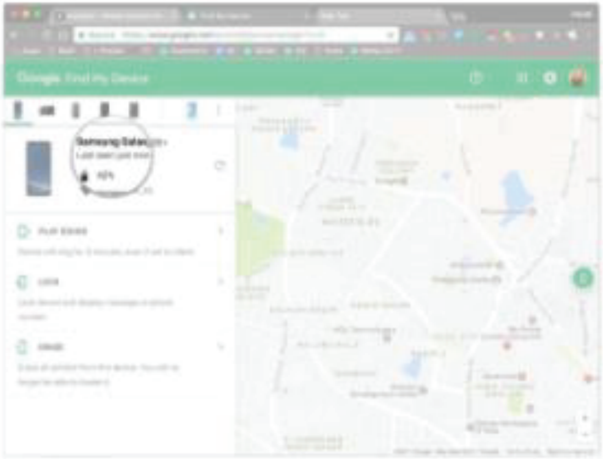
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	https://developers.google.com/android/reference/com/google/android/gms/location/LocationListener  Public Constructors public **MapView** (Context context) public **MapView** (Context context, AttributeSet attrs) public **MapView** (Context context, AttributeSet attrs, int defStyle) public **MapView** (Context context, GoogleMapOptions options) https://developers.google.com/android/reference/com/google/android/gms/maps/MapView

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public void **getMapAsync** (OnMapReadyCallback callback) Returns a non-null instance of the `GoogleMap`, ready to be used. Note that: • This method must be called from the main thread. • The callback will be executed in the main thread. • In the case where Google Play services is not installed on the user's device, the callback will not be triggered until the user installs it. • The `GoogleMap` object provided by the callback is non-null. **Parameters** **callback** The callback object that will be triggered when the map is ready to be used. public final void **onCreate** (Bundle savedInstanceState) You must call this method from the parent Activity/Fragment's corresponding method. https://developers.google.com/android/reference/com/google/android/gms/maps/MapView
[1G] presenting, via the interactive display of the first device, a second georeferenced map and a second set of one or more user-selectable symbols corresponding to a second set of one or more of the second devices, wherein the second set of symbols	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of: presenting, via the interactive display of the first device, a second georeferenced map and a second set of one or more user-selectable symbols corresponding to a second set of one or more of the second devices, wherein the second set of symbols are positioned on the second georeferenced map at respective positions corresponding to the locations of the second set of second devices, and wherein the second georeferenced map data relate positions on the second georeferenced map to spatial coordinates. **Regarding Find My Device** and Android Device Manager, the Accused Products present the user with a second and/or updated map display on the display. The second and/or updated map comprises a first symbol positioned on the map and corresponding to the first device. The second and/or updated map comprises one or more second symbols positioned on the map and corresponding to one or more second

US9467838	Exemplary Supporting Evidence Regarding Accused Products
are positioned on the second georeferenced map at respective positions corresponding to the locations of the second set of second devices, and wherein the second georeferenced map data relate positions on the second georeferenced map to spatial coordinates;	devices. The map is interactive because the user may control the display of the map, e.g., pan, zoom, and/or effect change to the map in an otherwise interactive manner. The map is georeferenced for at least the reason that one or more symbols are associated with spatial locations, i.e., coordinates. The symbols are user-selectable because a user may touch the display to select the device associated with the symbol. For example, the user may input a touch selection directly on or near the portion of the display corresponding to the symbol's coordinates on the map to effect a selection of the device. In an alternative example, the user may input a selection event on the display that does not directly correspond to the symbol's coordinates, but which effects a selection of the device because the selection event is otherwise associated with the device corresponding to the symbol. **<u>Regarding Google Maps</u>**, Google Latitude, Google Plus, Google Hangouts, Google Messages, Google Allo, Google Duo, Google Chrome, and Android Messenger, the Accused Products present the user of a first device with a second and/or updated map display on the display. The second and/or updated map comprises a first symbol positioned on the map and corresponding to the first user or corresponding first device. The second and/or updated map comprises one or more second symbols positioned on the map and corresponding to one or more second users, contacts and/or corresponding second devices. The map is interactive because the user may control the display of the map, e.g., pan, zoom, and/or effect change to the map in an otherwise interactive manner. The map is georeferenced for at least the reason that one or more symbols are associated with spatial locations, i.e., coordinates. The symbols are user-selectable because a user may touch the display to select the user or device associated with the symbol. For example, the user may input a touch selection directly on or near the portion of the display corresponding to the symbol's coordinates on the map to effect a selection of the user or device. In an alternative example, the user may input a selection event on the display that does not directly correspond to the symbol's coordinates, but which effects a selection of the user or device because the selection event is otherwise associated with the user or device corresponding to the symbol. **<u>Exemplary Support for Find My Device</u>**:

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 8:00 Find My Device SIGN OUT Birkbeck, University of London Russell Square Senate House Library The British Museum Google Rachel's phone Last seen just now 100% FreeWiFi PLAY SOUND LOCK ERASE 8:00 Find My Device St James's Palace Clarence House Lancaster House Google Rachel's watch Last seen just now 86% https://play.google.com/store/apps/details?id=com.google.android.apps.adm&hl=en

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	Once you're signed in to Find My Device, you'll see a map with your current location as well as the make and model of your phone, and two options — Play Sound, and Enable Lock & Erase. Hitting the latter option will allow you to start using the Lock and Erase functions. If you've signed into more than one phone, you can select a particular device by browsing the list at the top of the screen. 1. Open **Find My Device** from your home screen or app drawer. 2. Select your phone from the **list of devices at the top of the screen**. 3. See if your phone is **discoverable**.  Find My Device Samsung Galaxy S8+ Samsung In your hand ENABLE LOCK & ERASE https://www.androidcentral.com/find-my-device

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to locate your phone over the internet If you've lost your phone, you can remotely locate it through the Find My Device website. You'll need to sign in to the Google account that was used to set up Find My Device. It takes a few seconds, but the service should be able to track your phone. Alternatively, you can also do a Google search for "find my phone" to locate your handset. 1. Head to the **Find My Device website**. 2. Sign in to your **Google account**. 3. Check if your device is **visible**.  https://www.androidcentral.com/find-my-device **<u>Exemplary Support for Google Maps</u>**:

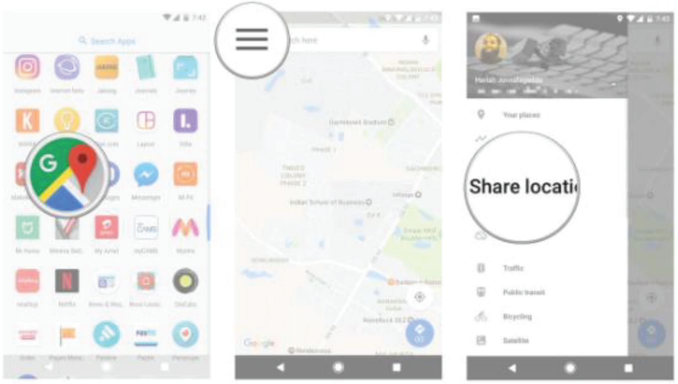
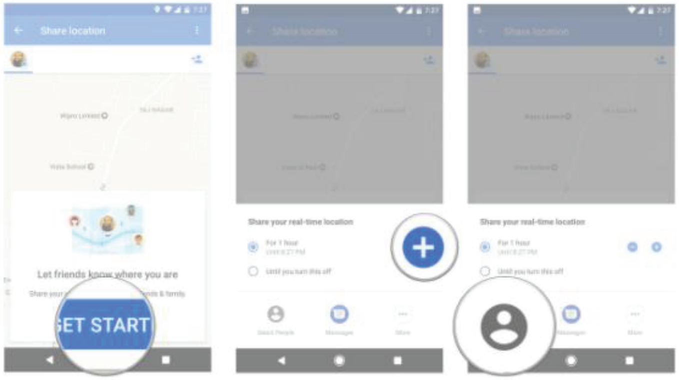
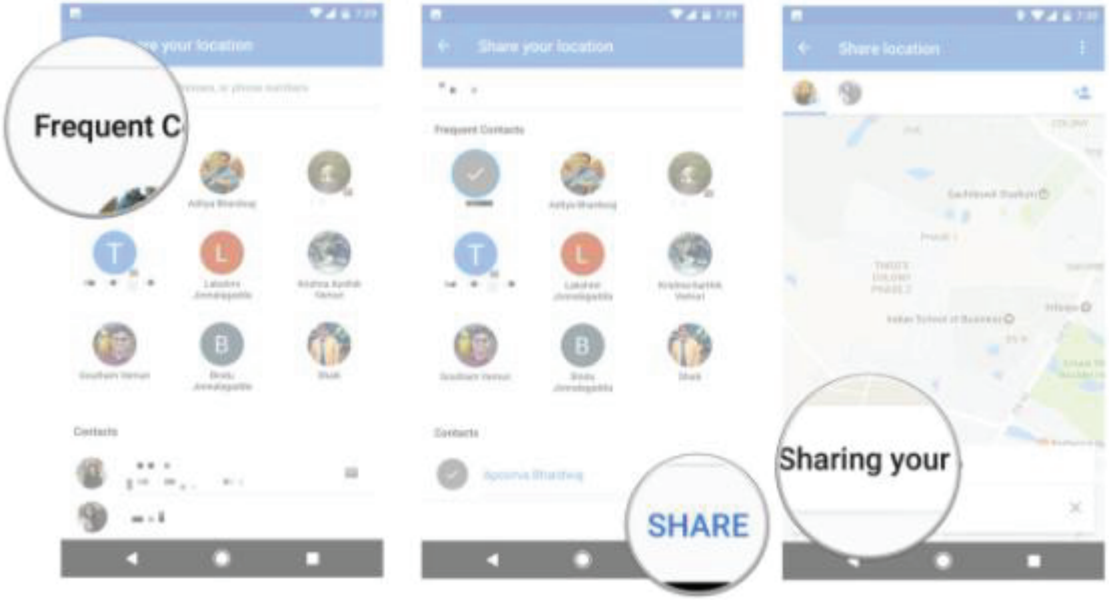
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to share your location in Google Maps 1. Open **Google Maps** from the app drawer or the home screen. 2. Tap the **hamburger menu** (the three horizontal lines) on the top left corner of the screen. 3. Select **Share location**.  4. Tap **Get Started**. 5. Use the **+** icon to select a time period or select the **Until you turn this off** setting to share your location indefinitely. 6. Tap **Select People**.  Share location Let friends know where you are GET START Share your real-time location For 1 hour Until you turn this off https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	7. You'll see a list of your frequent contacts at the top, along with a full list of contacts. **Pick the contacts** by tapping their name. 8. Once you've selected the contacts you want to share your location to, tap **Share**. 9. You'll see a message saying that the selected contact can view your location.  Frequent C Share your location Frequent Contacts Contacts SHARE Share location Sharing your https://www.androidcentral.com/how-share-location-google-maps

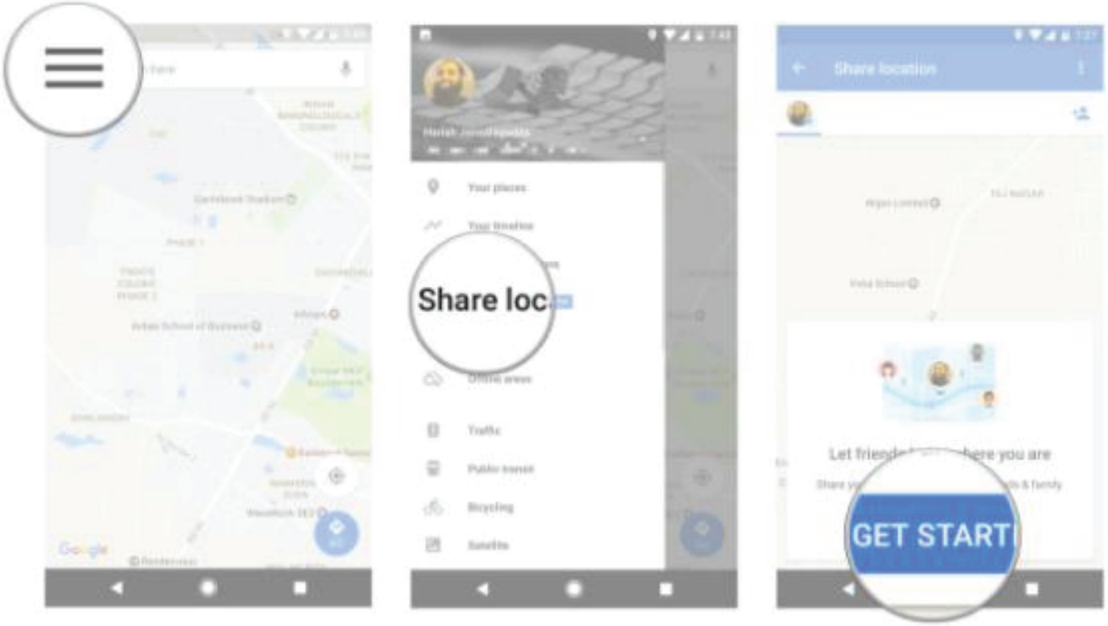
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to create a shareable link You can also create a link and use it to share your location easily. Here's how to do it: 1. Tap the **hamburger menu** on the top left corner of the screen. 2. Select **Share location**. 3. Tap **Get Started**.  Share loc Share location Let friends ... where you are GET START https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

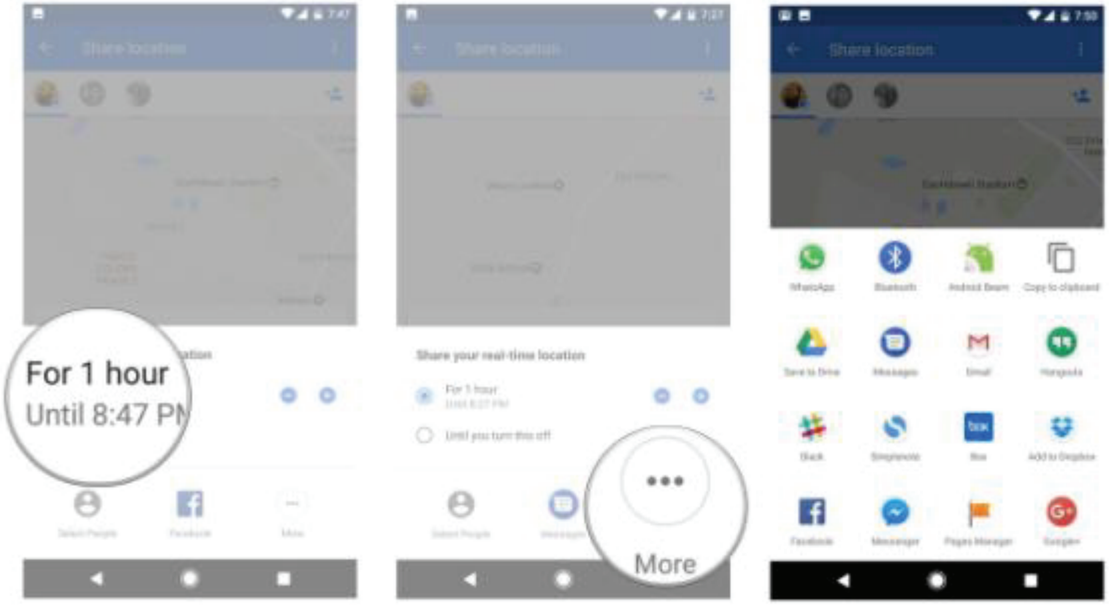
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	4. Select the **amount of time** you want to share your location. 5. Tap **More**. 6. Select **your app of choice** to create and send a unique URL that broadcasts your current location. You can email it, send the link via Messenger, or even tweet it to the intended recipient.  Share location For 1 hour Until 8:47 PM Share your real-time location Until you turn this off More https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

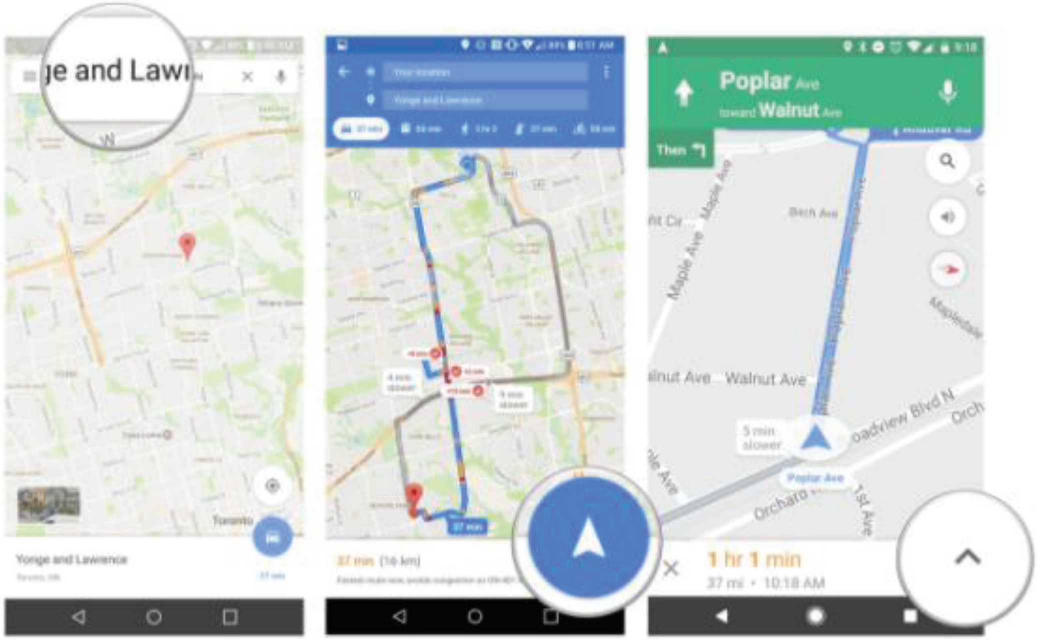
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to share your navigation directions while you walk, drive or transit One of the best ways to share your location with someone is during a drive, walk or transit. If you're meeting a friend or family member somewhere, or navigating towards their home, Google Maps lets you share your location with them for the duration of the trip. It's magic! 1. In the **search bar** enter your destination. 2. Pick your navigation type (drive, transit, walk) and press the blue **navigate button**. 3. Tap the arrow next to the **time-to-destination** number at the bottom of the screen.  ge and Lawr Yonge and Lawrence Toronto 37 min (16 km) 4 min slower 9 min slower 37 min Poplar Ave toward Walnut Ave Then Maple Ave Birch Ave Walnut Ave 5 min slower Broadview Blvd N Poplar Ave Orchard 1st Ave 1 hr 1 min 37 mi • 10:18 AM https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

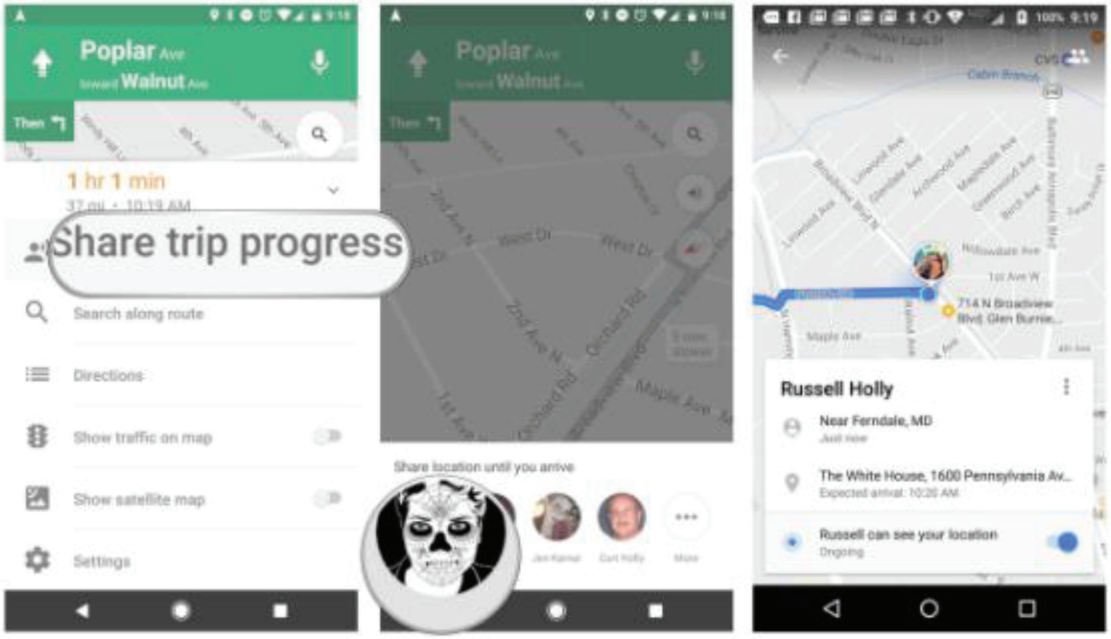
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	4. Tap **Share trip progress**. 5. Choose one or more contacts to share trip progress.  Poplar Ave toward Walnut Ave Then 1 hr 1 min 37 mi · 10:19 AM Share trip progress Search along route Directions Show traffic on map Show satellite map Settings Share location until you arrive More Russell Holly Near Ferndale, MD Just now The White House, 1600 Pennsylvania Av... Expected arrival: 10:20 AM Russell can see your location Ongoing 714 N Broadview Blvd, Glen Burnie... You can also stop sharing your location with someone before a trip ends. https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

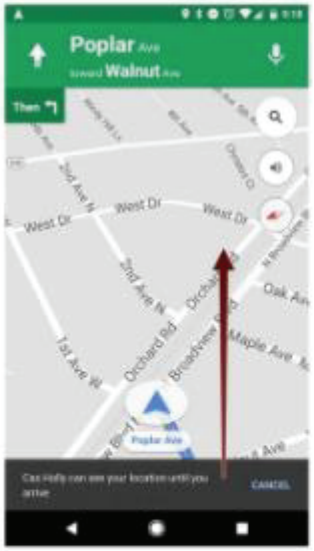
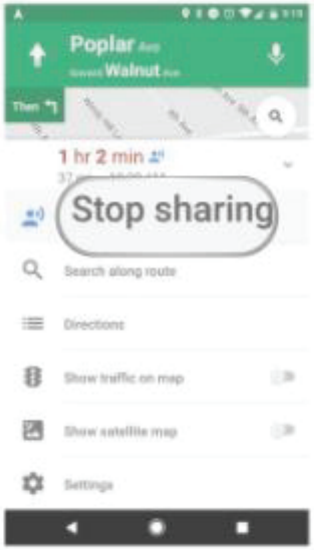
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	1. Tap the arrow next to the **time-to-destination** number at the bottom of the screen. 2. Tap **Stop sharing.**  Poplar Ave toward Walnut Ave Then West Dr 2nd Ave N Orchard Rd Broadview Oak Ave Maple Ave 1st Ave W Poplar Ave CANCEL Poplar Ave toward Walnut Ave Then 1 hr 2 min Stop sharing Search along route Directions Show traffic on map Show satellite map Settings That's it! Are you excited that location sharing is back in Google Maps? How often do you use the feature? https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

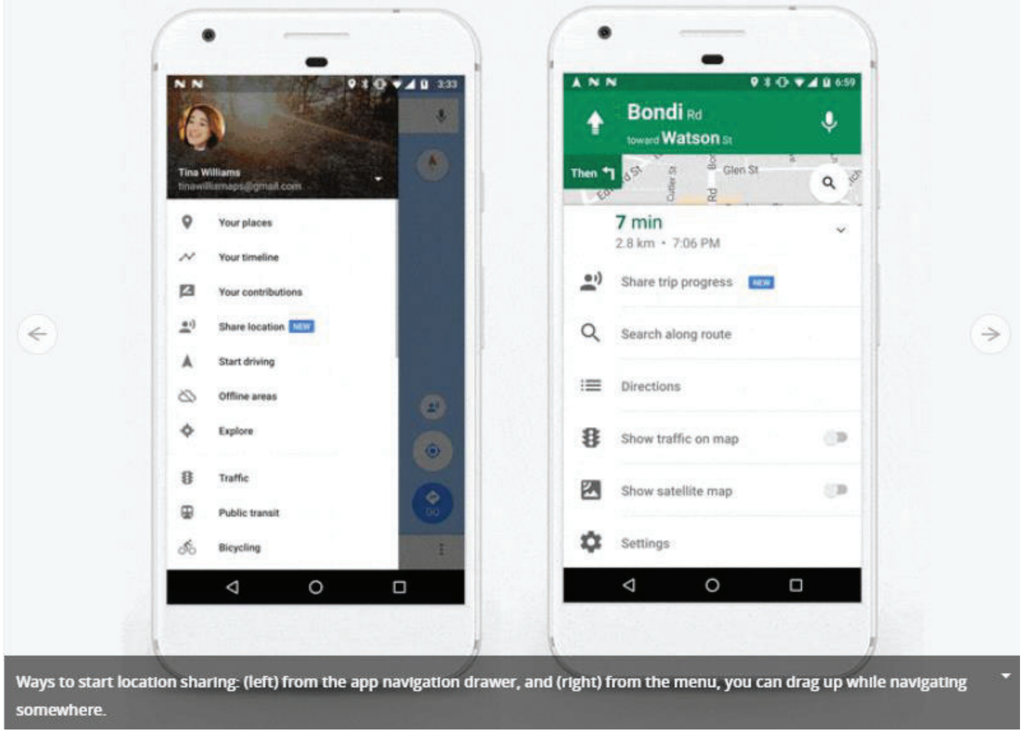
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Tina Williams tinawilliamsmaps@gmail.com Your places Your timeline Your contributions Share location NEW Start driving Offline areas Explore Traffic Public transit Bicycling Bondi Rd toward Watson St Then Glen St 7 min 2.8 km • 7:06 PM Share trip progress NEW Search along route Directions Show traffic on map Show satellite map Settings Ways to start location sharing: (left) from the app navigation drawer, and (right) from the menu, you can drag up while navigating somewhere. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

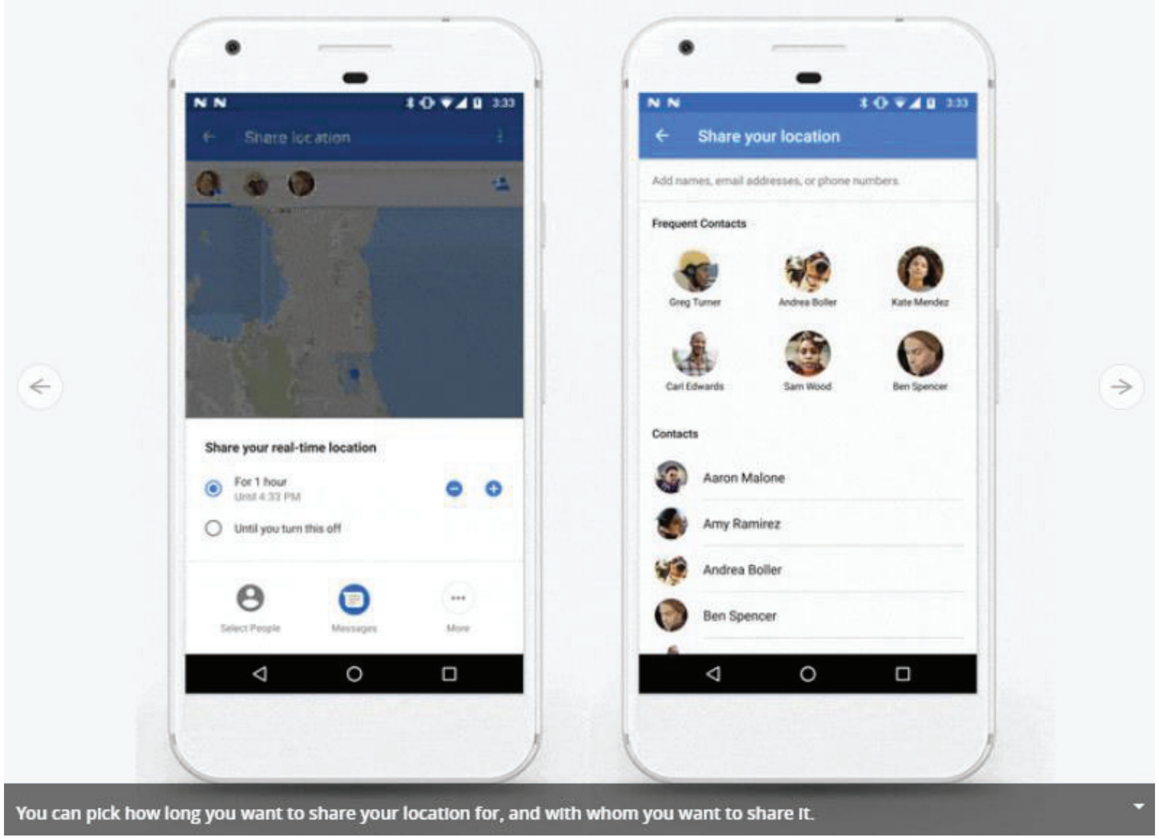
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Share location Share your real-time location For 1 hour Until 4:33 PM Until you turn this off Select People Messages More Share your location Add names, email addresses, or phone numbers Frequent Contacts Greg Turner Andrea Boller Kate Mendez Carl Edwards Sam Wood Ben Spencer Contacts Aaron Malone Amy Ramirez Andrea Boller Ben Spencer You can pick how long you want to share your location for, and with whom you want to share it. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

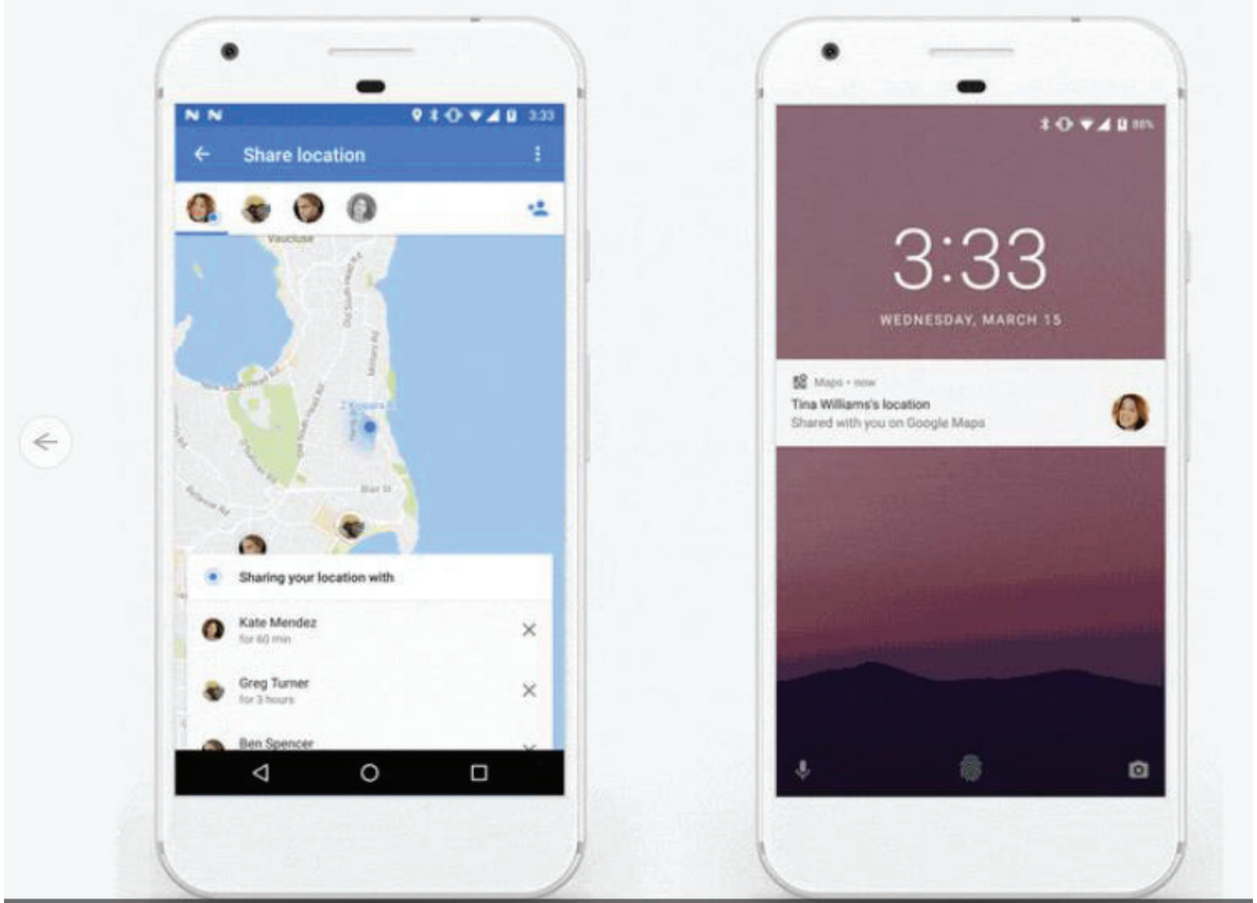
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Share location Sharing your location with Kate Mendez for 60 min Greg Turner for 3 hours Ben Spencer 3:33 WEDNESDAY, MARCH 15 Maps • now Tina Williams's location Shared with you on Google Maps People you've shared with will get a notification from Google Maps. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

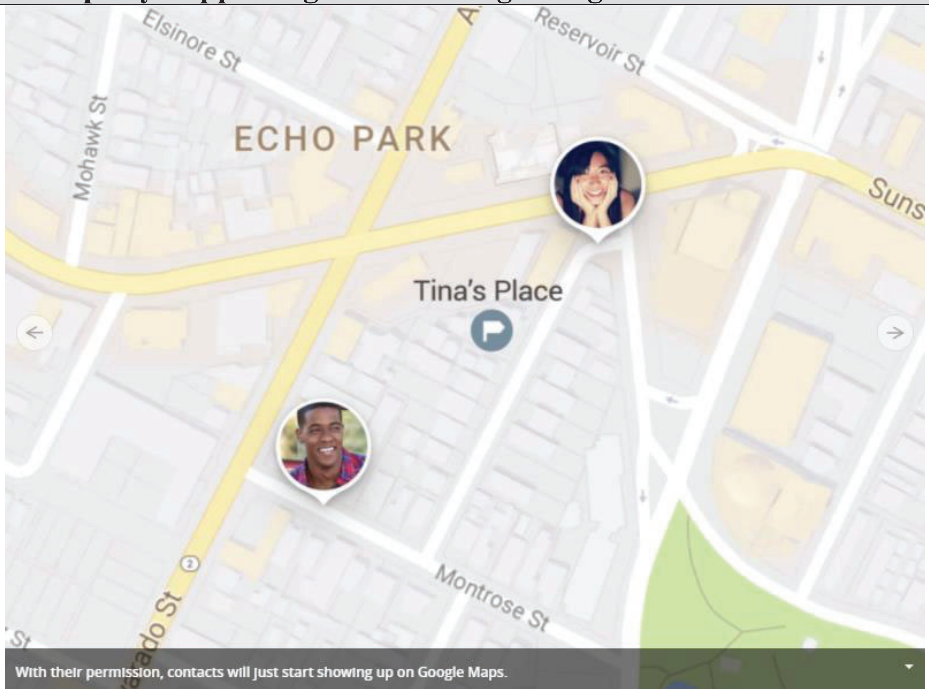
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Elsinore St Reservoir St Mohawk St ECHO PARK Suns Tina's Place Montrose St ado St With their permission, contacts will just start showing up on Google Maps. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/ **<u>Exemplary Find My Device Screenshots</u>**:

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

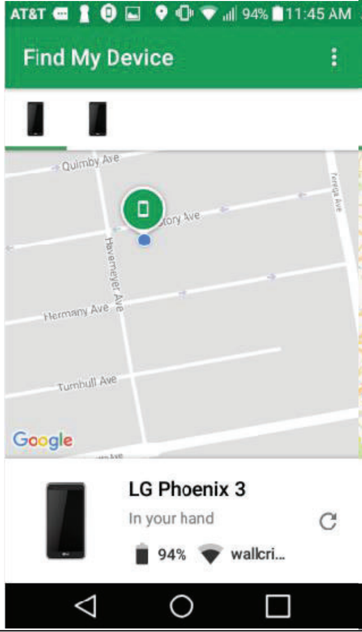
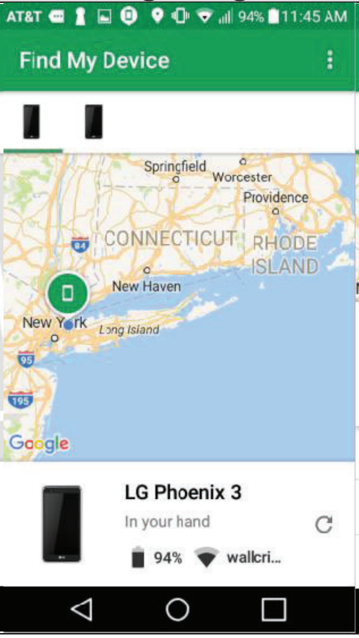
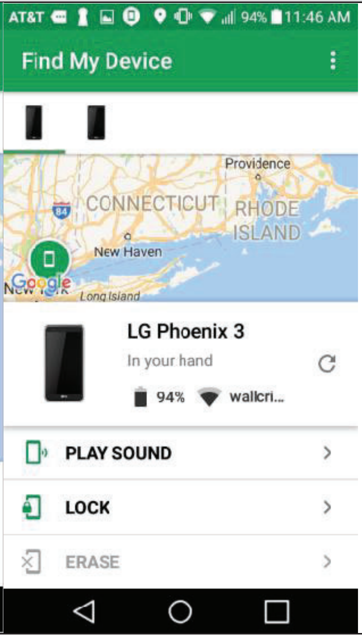
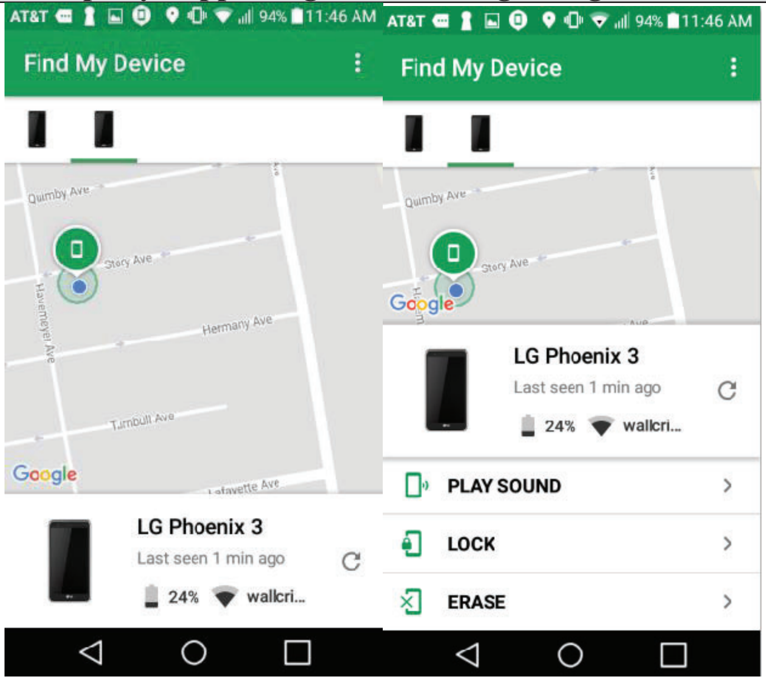
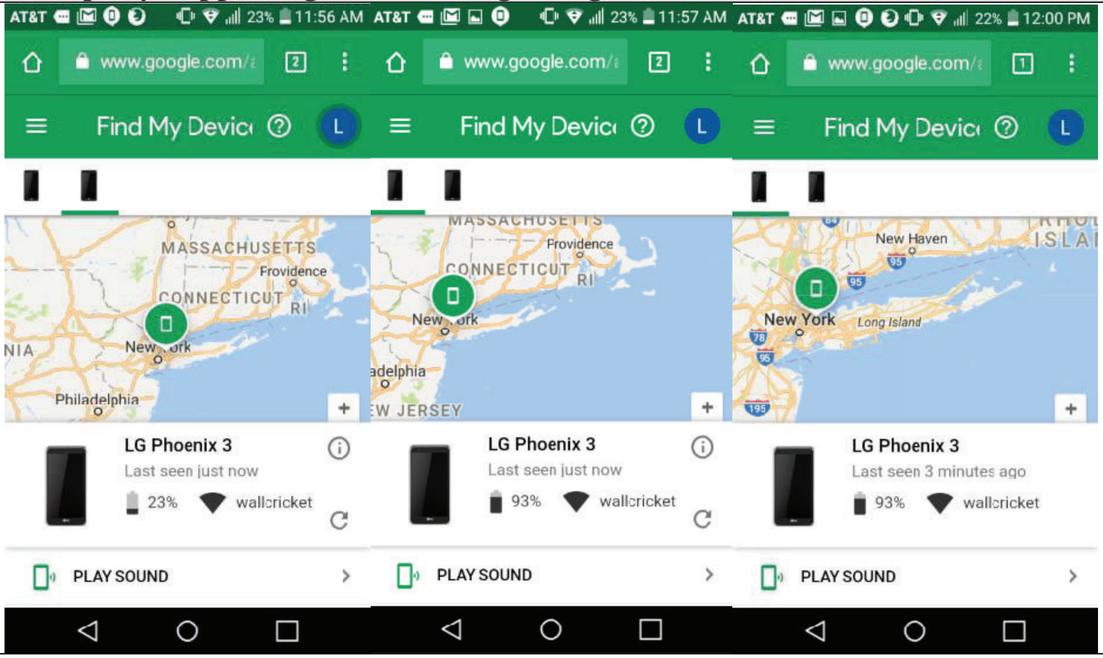
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	  

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 94% 11:46 AM Find My Device Quimby Ave Story Ave Havemeyer Ave Hermany Ave Turnbull Ave Lafayette Ave Google LG Phoenix 3 Last seen 1 min ago 24% wallcri... AT&T 94% 11:46 AM Find My Device Quimby Ave Story Ave Google LG Phoenix 3 Last seen 1 min ago 24% wallcri... PLAY SOUND LOCK ERASE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 23% 11:56 AM www.google.com/ Find My Devic L MASSACHUSETTS Providence CONNECTICUT RI NIA New York Philadelphia LG Phoenix 3 Last seen just now 23% wallcricket PLAY SOUND AT&T 23% 11:57 AM www.google.com/ Find My Devic L MASSACHUSETTS Providence CONNECTICUT RI New York adelphia EW JERSEY LG Phoenix 3 Last seen just now 93% wallcricket PLAY SOUND AT&T 22% 12:00 PM www.google.com/ Find My Devic L New Haven ISLA New York Long Island LG Phoenix 3 Last seen 3 minutes ago 93% wallcricket PLAY SOUND

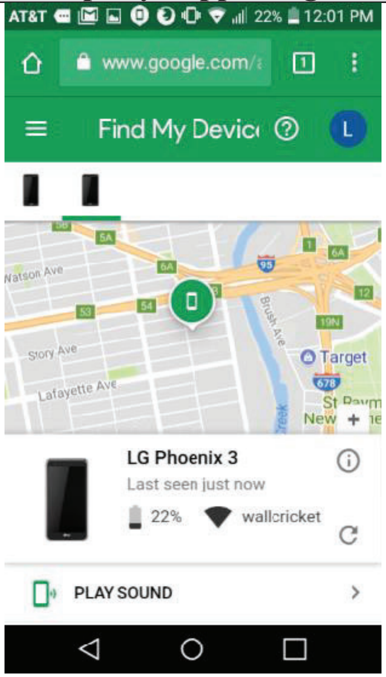
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 22% 12:01 PM www.google.com/ Find My Devic LG Phoenix 3 Last seen just now 22% wallcricket PLAY SOUND **<u>Exemplary Google Maps Screenshots</u>**:

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

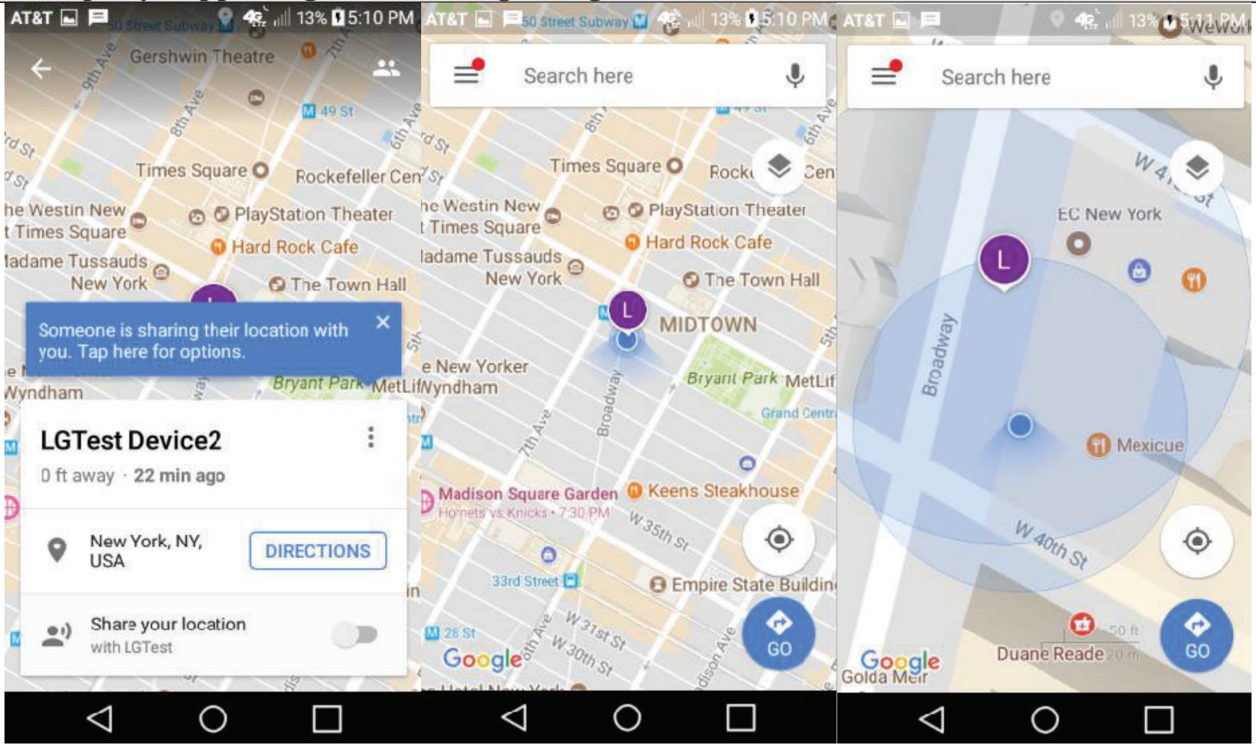
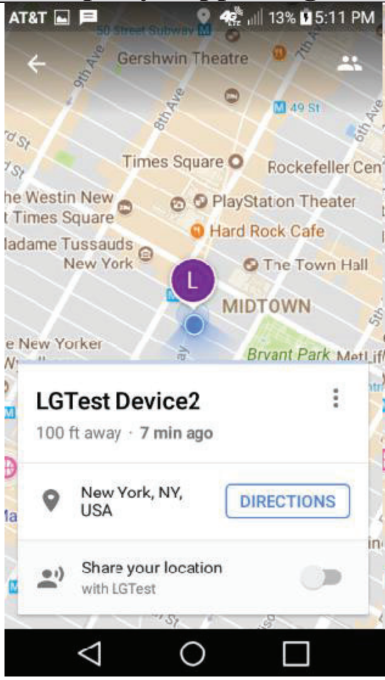
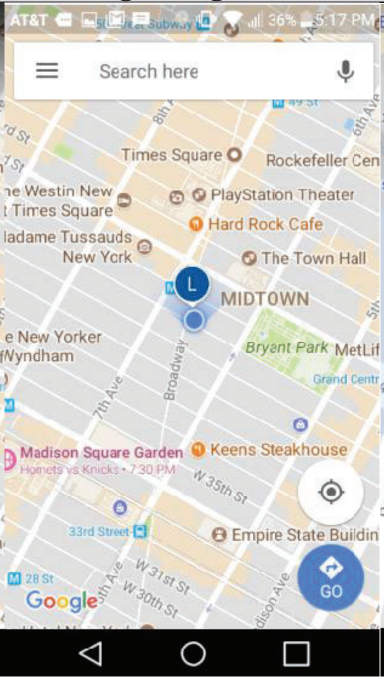
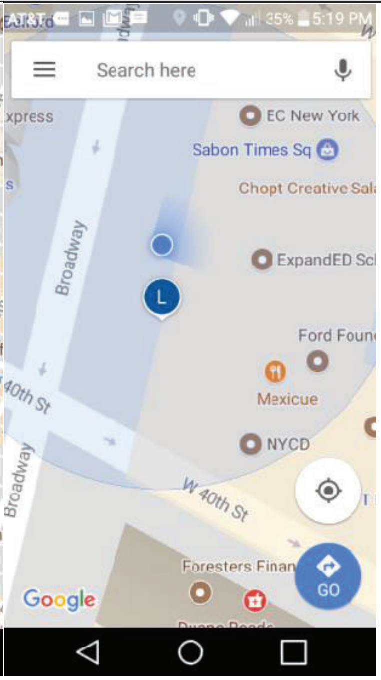
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Someone is sharing their location with you. Tap here for options. LGTest Device2 0 ft away · 22 min ago New York, NY, USA DIRECTIONS Share your location with LGTest AT&T 13% 5:10 PM 13% 5:11 PM Search here Times Square Rockefeller Cen PlayStation Theater Hard Rock Cafe The Town Hall MIDTOWN Bryant Park Madison Square Garden Keens Steakhouse Empire State Buildin Broadway EC New York Mexicue W 40th St Duane Reade GO Google

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 13% 5:11 PM Gershwin Theatre Times Square Rockefeller Cen PlayStation Theater Hard Rock Cafe The Town Hall MIDTOWN Bryant Park L LGTest Device2 100 ft away · 7 min ago New York, NY, USA DIRECTIONS Share your location with LGTest  AT&T 36% 5:17 PM Search here Times Square Rockefeller Cen PlayStation Theater Hard Rock Cafe The Town Hall MIDTOWN Bryant Park Madison Square Garden Keens Steakhouse Empire State Buildin L GO Google  AT&T 35% 5:19 PM Search here EC New York Sabon Times Sq Chopt Creative Sal ExpandED Sc Ford Foun Mexicue NYCD Broadway 40th St W 40th St Foresters Finan L GO Google

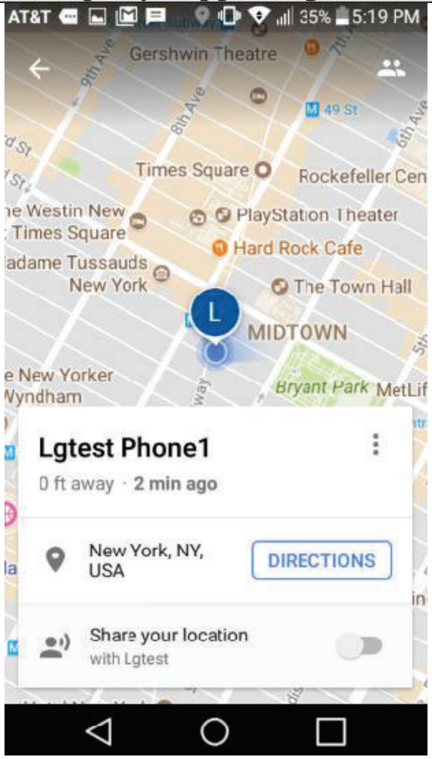
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 35% 5:19 PM Gershwin Theatre 9th Ave 8th Ave 7th Ave 6th Ave 49 St Times Square Rockefeller Cen e Westin New Times Square PlayStation Theater Hard Rock Cafe adame Tussauds New York The Town Hall L MIDTOWN e New Yorker Wyndham Bryant Park MetLif DIRECTIONS Lgtest Phone1 0 ft away · 2 min ago New York, NY, USA Share your location with Lgtest **<u>Exemplary Source Code</u>**: The above functionality is performed at least in part by the following publicly available source code and/or source code that invokes or is invoked by the following source code (or a substantially similar copy compiled and loaded onto the Accused Products by ZTE):

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

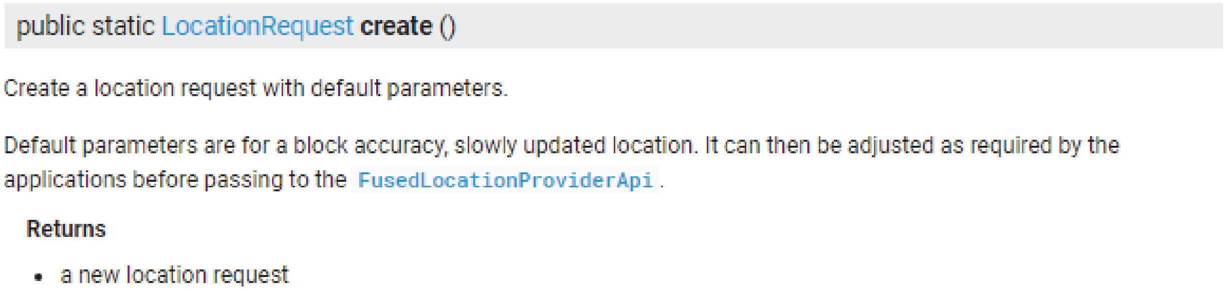
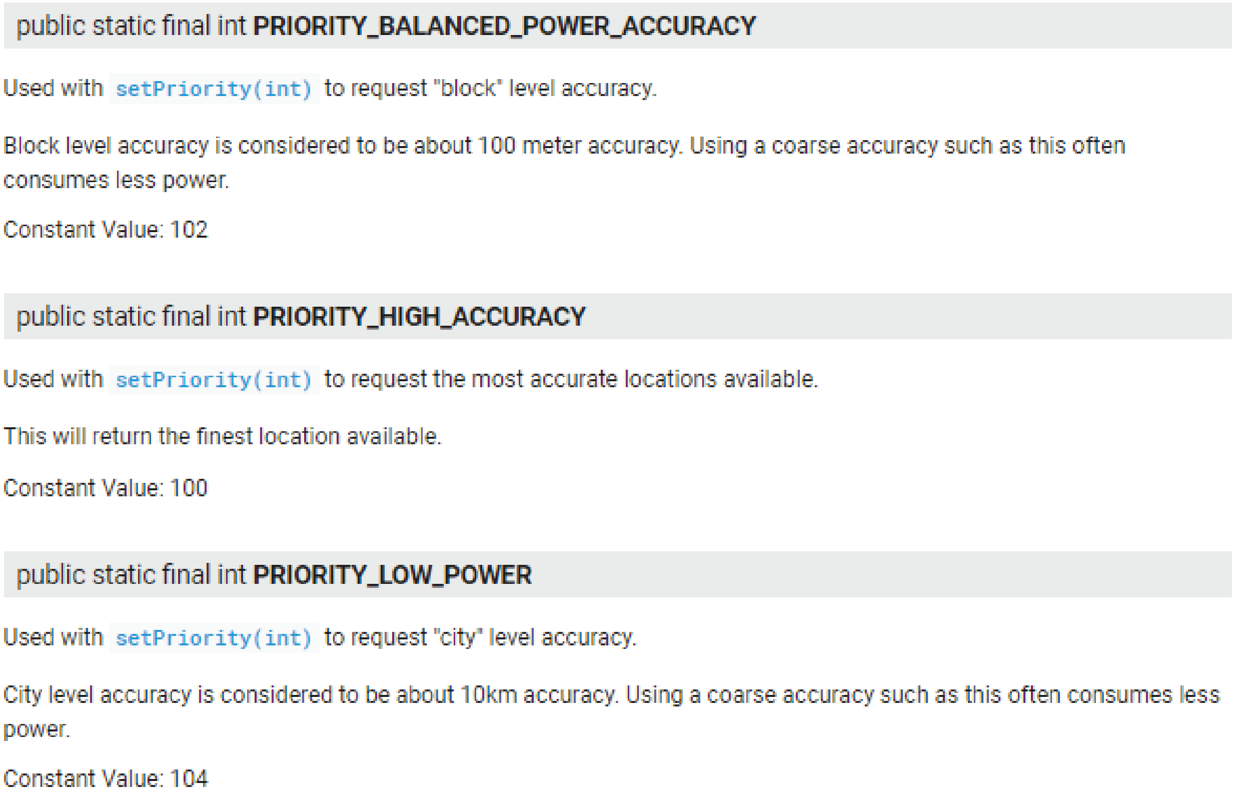
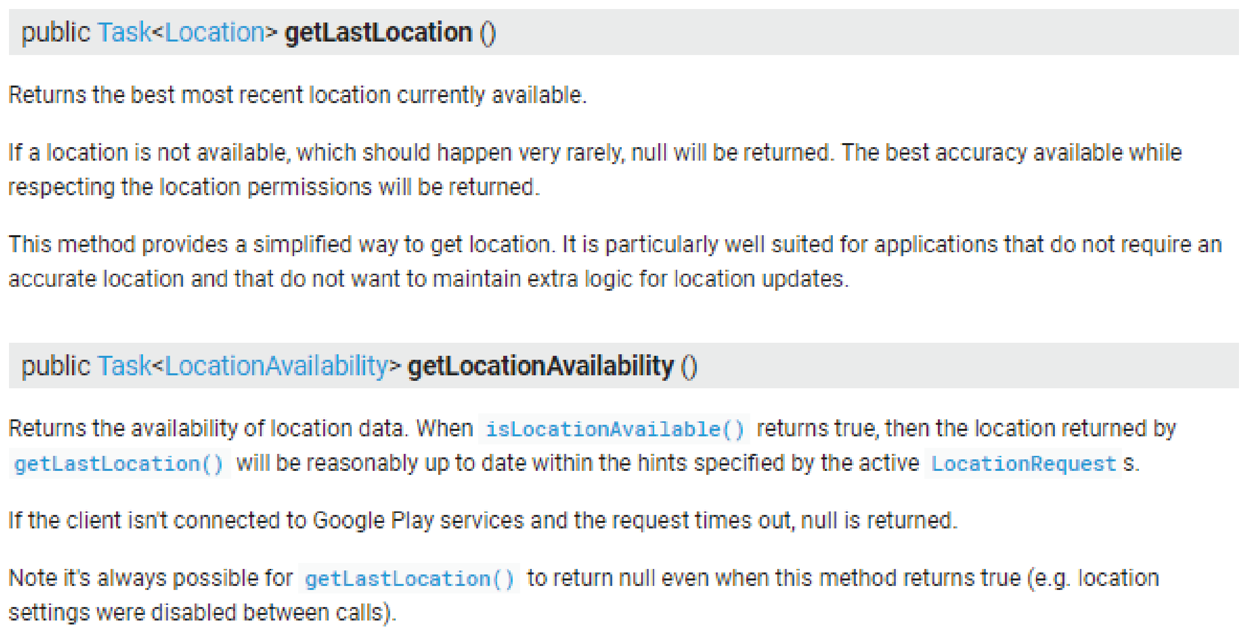
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public static LocationRequest **create** () Create a location request with default parameters. Default parameters are for a block accuracy, slowly updated location. It can then be adjusted as required by the applications before passing to the `FusedLocationProviderApi`. **Returns** • a new location request https://developers.google.com/android/reference/com/google/android/gms/location/LocationRequest  public static final int **PRIORITY_BALANCED_POWER_ACCURACY** Used with `setPriority(int)` to request "block" level accuracy. Block level accuracy is considered to be about 100 meter accuracy. Using a coarse accuracy such as this often consumes less power. Constant Value: 102 public static final int **PRIORITY_HIGH_ACCURACY** Used with `setPriority(int)` to request the most accurate locations available. This will return the finest location available. Constant Value: 100 public static final int **PRIORITY_LOW_POWER** Used with `setPriority(int)` to request "city" level accuracy. City level accuracy is considered to be about 10km accuracy. Using a coarse accuracy such as this often consumes less power. Constant Value: 104 https://developers.google.com/android/reference/com/google/android/gms/location/LocationRequest

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public Task<Location> **getLastLocation** () Returns the best most recent location currently available. If a location is not available, which should happen very rarely, null will be returned. The best accuracy available while respecting the location permissions will be returned. This method provides a simplified way to get location. It is particularly well suited for applications that do not require an accurate location and that do not want to maintain extra logic for location updates. public Task<LocationAvailability> **getLocationAvailability** () Returns the availability of location data. When `isLocationAvailable()` returns true, then the location returned by `getLastLocation()` will be reasonably up to date within the hints specified by the active `LocationRequest` s. If the client isn't connected to Google Play services and the request times out, null is returned. Note it's always possible for `getLastLocation()` to return null even when this method returns true (e.g. location settings were disabled between calls). https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient

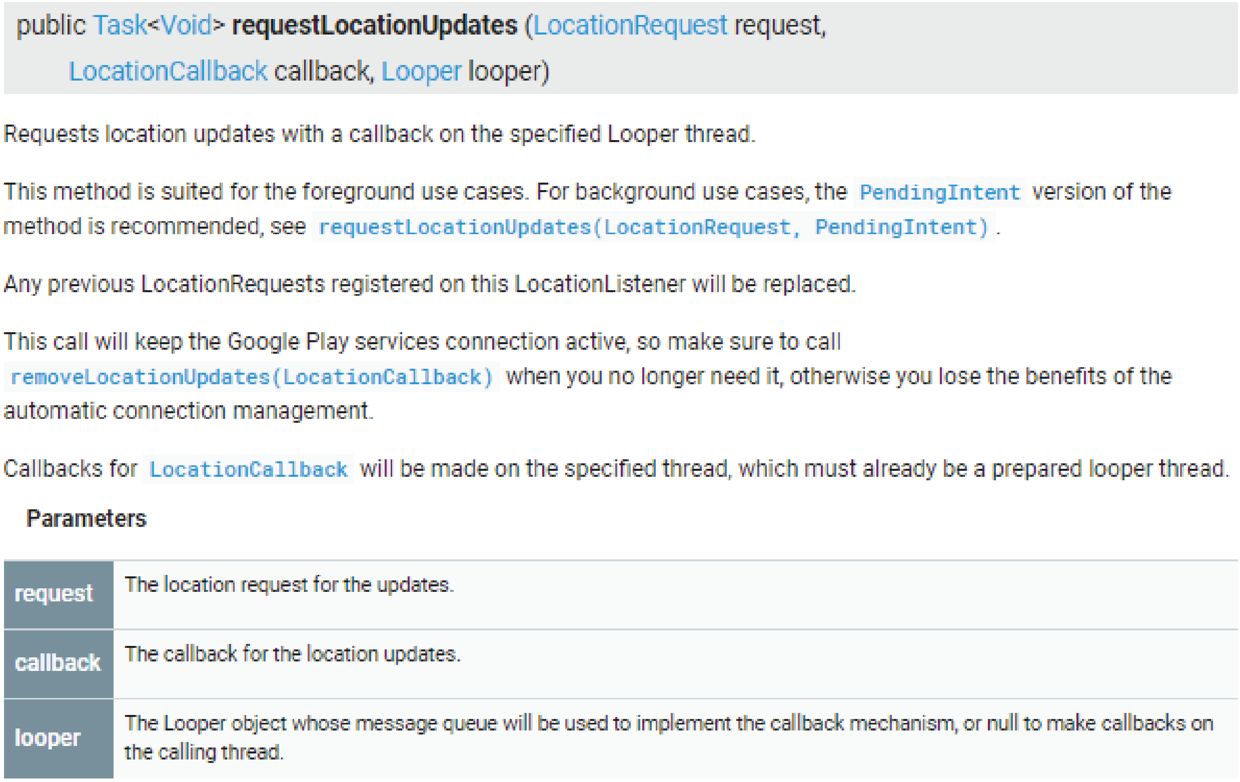
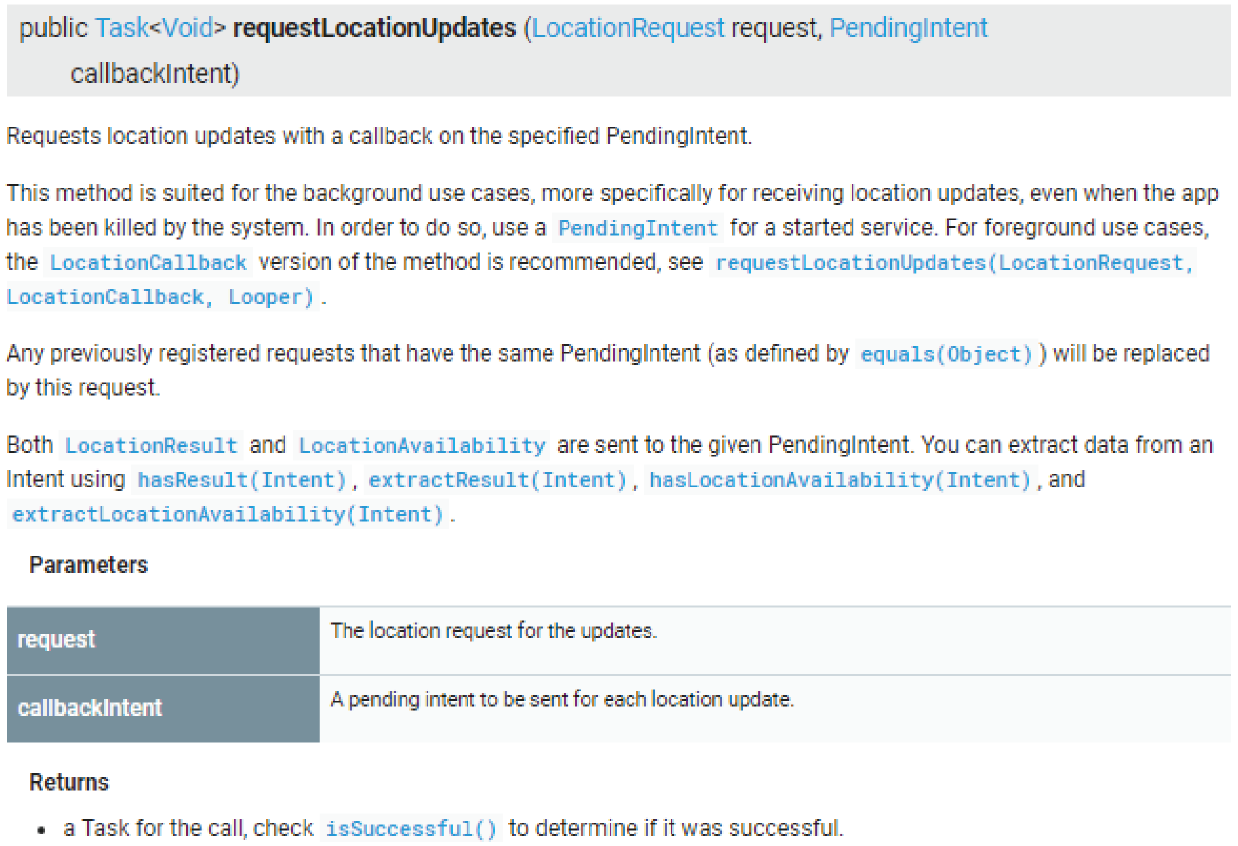
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public Task<Void> **requestLocationUpdates** (LocationRequest request, LocationCallback callback, Looper looper) Requests location updates with a callback on the specified Looper thread. This method is suited for the foreground use cases. For background use cases, the `PendingIntent` version of the method is recommended, see `requestLocationUpdates(LocationRequest, PendingIntent)`. Any previous LocationRequests registered on this LocationListener will be replaced. This call will keep the Google Play services connection active, so make sure to call `removeLocationUpdates(LocationCallback)` when you no longer need it, otherwise you lose the benefits of the automatic connection management. Callbacks for `LocationCallback` will be made on the specified thread, which must already be a prepared looper thread. **Parameters** **request** The location request for the updates. **callback** The callback for the location updates. **looper** The Looper object whose message queue will be used to implement the callback mechanism, or null to make callbacks on the calling thread. https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public Task<Void> **requestLocationUpdates** (LocationRequest request, PendingIntent callbackIntent) Requests location updates with a callback on the specified PendingIntent. This method is suited for the background use cases, more specifically for receiving location updates, even when the app has been killed by the system. In order to do so, use a `PendingIntent` for a started service. For foreground use cases, the `LocationCallback` version of the method is recommended, see `requestLocationUpdates(LocationRequest, LocationCallback, Looper)`. Any previously registered requests that have the same PendingIntent (as defined by `equals(Object)`) will be replaced by this request. Both `LocationResult` and `LocationAvailability` are sent to the given PendingIntent. You can extract data from an Intent using `hasResult(Intent)`, `extractResult(Intent)`, `hasLocationAvailability(Intent)`, and `extractLocationAvailability(Intent)`. **Parameters** **request** The location request for the updates. **callbackIntent** A pending intent to be sent for each location update. **Returns** - a Task for the call, check `isSuccessful()` to determine if it was successful. https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient

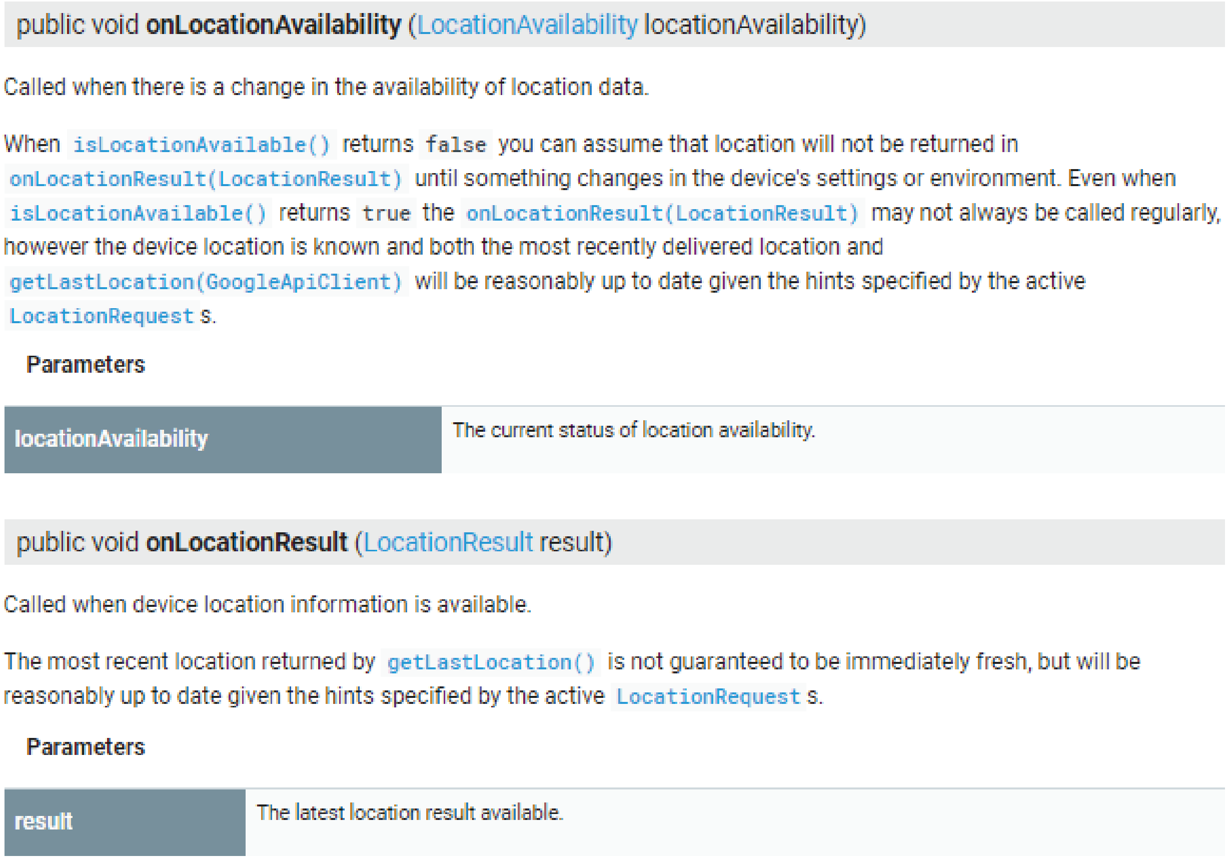
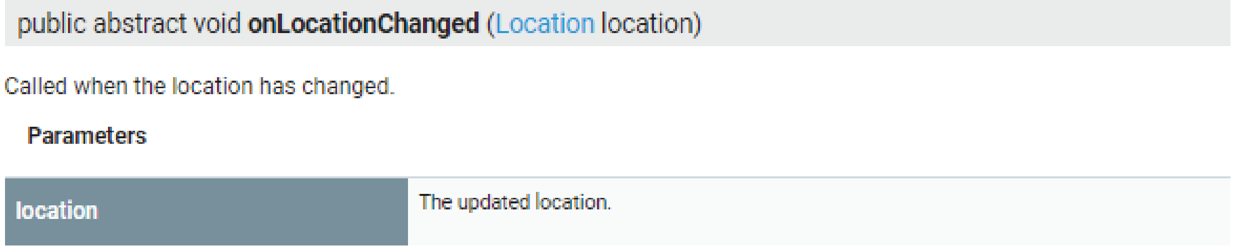
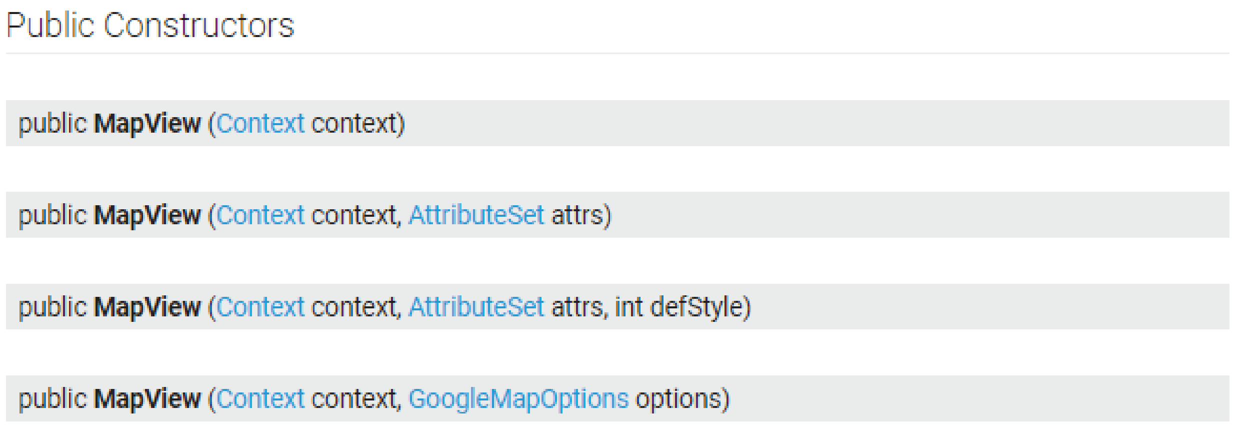
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public void **onLocationAvailability** (LocationAvailability locationAvailability) Called when there is a change in the availability of location data. When `isLocationAvailable()` returns `false` you can assume that location will not be returned in `onLocationResult(LocationResult)` until something changes in the device's settings or environment. Even when `isLocationAvailable()` returns `true` the `onLocationResult(LocationResult)` may not always be called regularly, however the device location is known and both the most recently delivered location and `getLastLocation(GoogleApiClient)` will be reasonably up to date given the hints specified by the active `LocationRequest` s. **Parameters** **locationAvailability** The current status of location availability. public void **onLocationResult** (LocationResult result) Called when device location information is available. The most recent location returned by `getLastLocation()` is not guaranteed to be immediately fresh, but will be reasonably up to date given the hints specified by the active `LocationRequest` s. **Parameters** **result** The latest location result available. https://developers.google.com/android/reference/com/google/android/gms/location/LocationCallback  public abstract void **onLocationChanged** (Location location) Called when the location has changed. **Parameters** **location** The updated location.

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	https://developers.google.com/android/reference/com/google/android/gms/location/LocationListener  Public Constructors public **MapView** (Context context) public **MapView** (Context context, AttributeSet attrs) public **MapView** (Context context, AttributeSet attrs, int defStyle) public **MapView** (Context context, GoogleMapOptions options) https://developers.google.com/android/reference/com/google/android/gms/maps/MapView

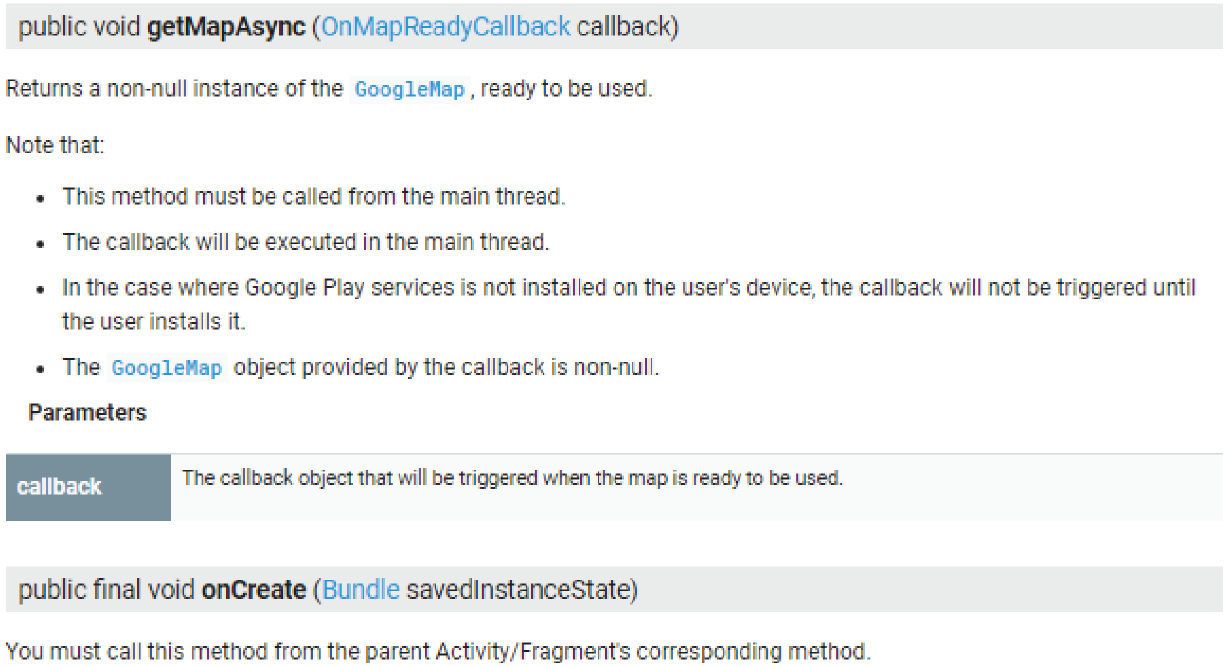
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public void **getMapAsync** (OnMapReadyCallback callback) Returns a non-null instance of the `GoogleMap`, ready to be used. Note that: • This method must be called from the main thread. • The callback will be executed in the main thread. • In the case where Google Play services is not installed on the user's device, the callback will not be triggered until the user installs it. • The `GoogleMap` object provided by the callback is non-null. **Parameters** callback: The callback object that will be triggered when the map is ready to be used. public final void **onCreate** (Bundle savedInstanceState) You must call this method from the parent Activity/Fragment's corresponding method. https://developers.google.com/android/reference/com/google/android/gms/maps/MapView
[1H] and identifying user interaction with the interactive display selecting one or more of the second set of user-selectable symbols corresponding to one or more of the second devices and positioned on the second georeferenced map and	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of: identifying user interaction with the interactive display selecting one or more of the second set of user-selectable symbols corresponding to one or more of the second devices and positioned on the second georeferenced map and user interaction with the display specifying an action and, based thereon, sending third data to the selected one or more second devices via the first server. **Regarding Find My Device** and Android Device Manager, the Accused Products are configured to allow a user of a first device to interact with the display, to select a device corresponding to a symbol, and to select an action to be performed, such as: play a sound on the second device, put the second device into a lost mode, and erase the second device. Selection of one of the aforementioned actions results in sending data from the first device to a server and then sending data from the server to the second device.

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

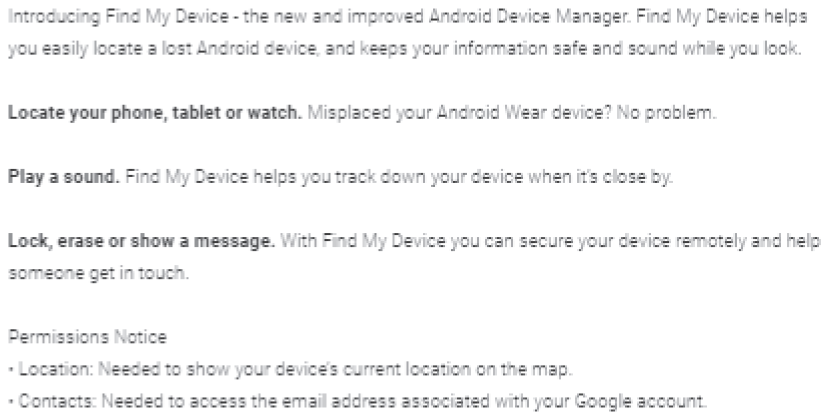
US9467838	Exemplary Supporting Evidence Regarding Accused Products
user interaction with the display specifying an action and, based thereon, sending third data to the selected one or more second devices via the first server.	**Regarding Google Maps**, Google Latitude, Google Plus, Google Hangouts, Google Messages, and Android Messenger, the Accused Products are configured to allow a user of the first device to interact with the display, to select a user, contact, or device, and to select an action to be performed, such as: sending a message, initiating a call, initiating a data conference, sharing a location, stop sharing a location, block a user from location sharing, sending a location, requesting a location, or sending other data. When an action is specified, data is sent from the first device to the second device via a server. **Exemplary Support for Find My Device**:  Introducing Find My Device - the new and improved Android Device Manager. Find My Device helps you easily locate a lost Android device, and keeps your information safe and sound while you look. **Locate your phone, tablet or watch.** Misplaced your Android Wear device? No problem. **Play a sound.** Find My Device helps you track down your device when it's close by. **Lock, erase or show a message.** With Find My Device you can secure your device remotely and help someone get in touch. Permissions Notice • Location: Needed to show your device's current location on the map. • Contacts: Needed to access the email address associated with your Google account. https://play.google.com/store/apps/details?id=com.google.android.apps.adm&hl=en

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

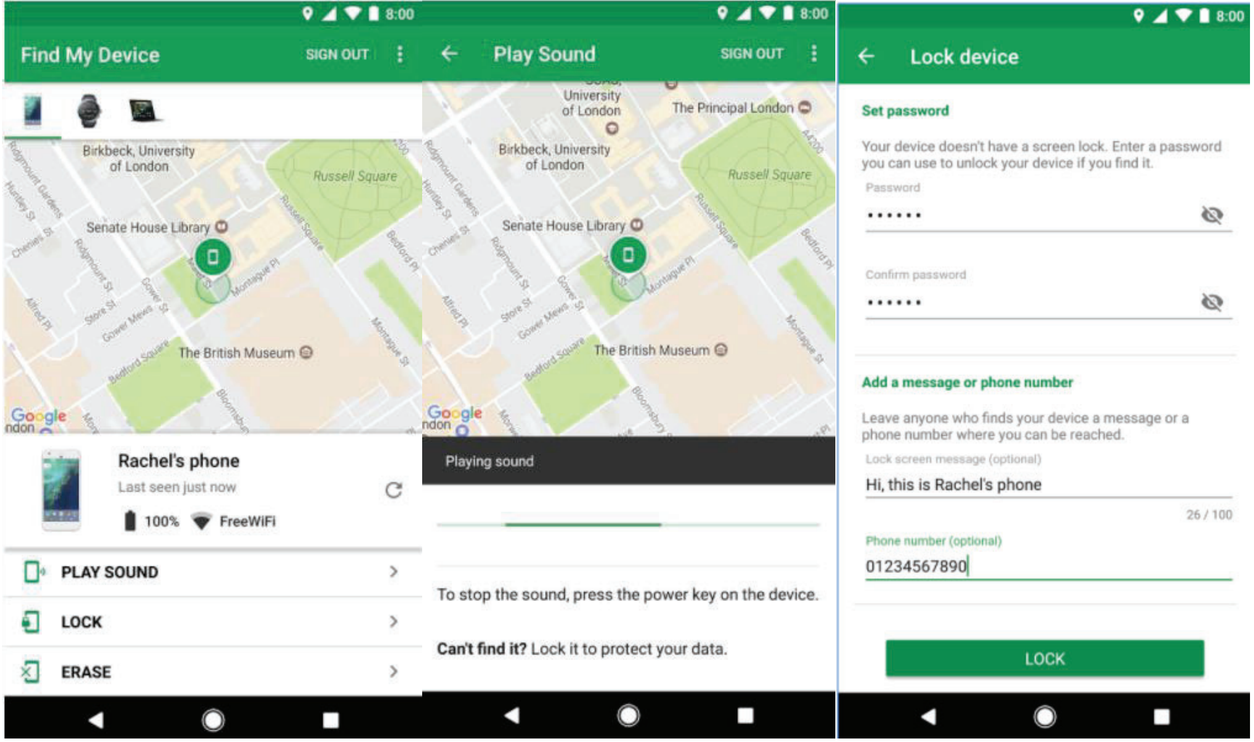
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 8:00 Find My Device SIGN OUT Birkbeck, University of London Russell Square Senate House Library The British Museum Google PLAY SOUND LOCK ERASE Play Sound University of London The Principal London Lock device LOCK Rachel's phone Last seen just now 100% FreeWiFi Playing sound To stop the sound, press the power key on the device. **Can't find it?** Lock it to protect your data. **Set password** Your device doesn't have a screen lock. Enter a password you can use to unlock your device if you find it. Password: •••••• Confirm password: •••••• **Add a message or phone number** Leave anyone who finds your device a message or a phone number where you can be reached. Lock screen message (optional): Hi, this is Rachel's phone 26 / 100 Phone number (optional): 01234567890 https://play.google.com/store/apps/details?id=com.google.android.apps.adm&hl=en

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

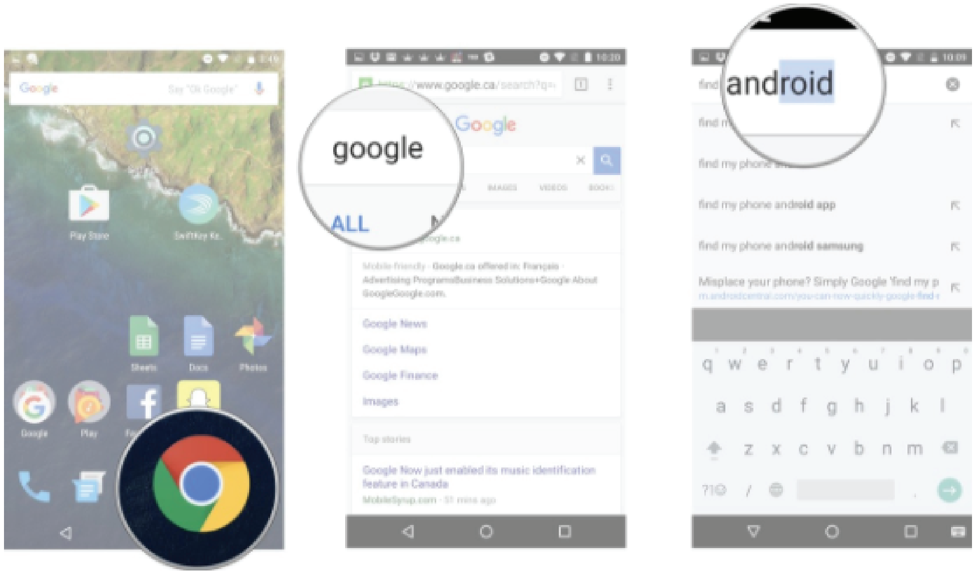
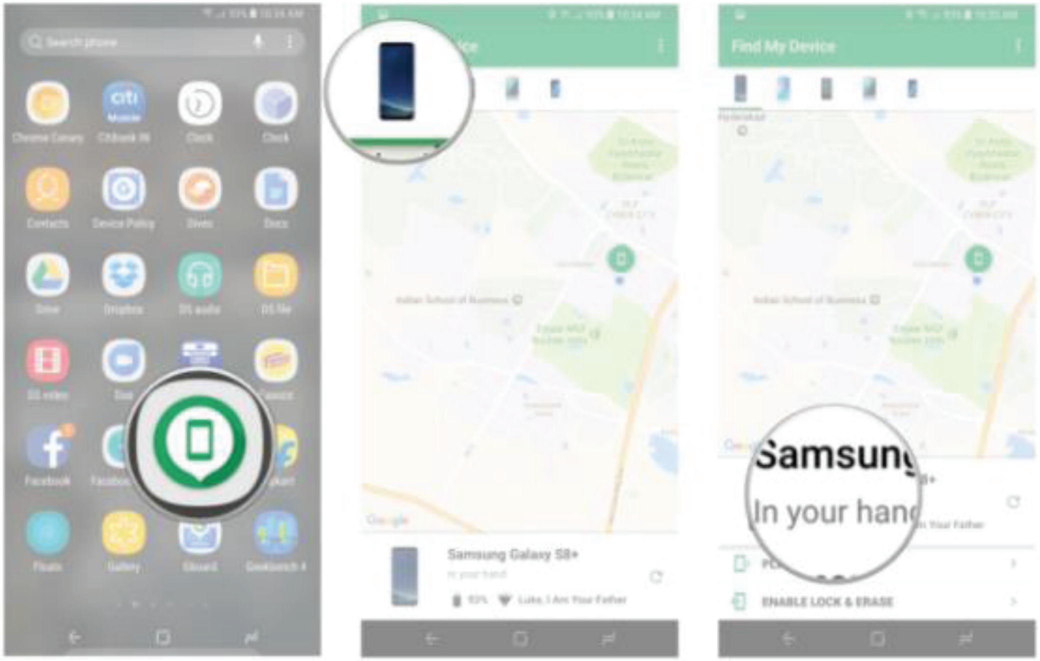
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to locate your phone with Google Should you happen to lose your phone, you can locate its whereabouts by logging into your Google account from any computer or even from another phone. 1. Launch a **web browser** from a phone, tablet, or computer. 2. Navigate to **Google** if it is not your default search engine or home page. 3. Type **find my phone android** in the Google search bar.  google ALL Google android find my phone android app find my phone android samsung 4. Tap on **Find My Device** (usually the first option in the search). 5. Enter your **email address** and **password** just as though you were checking your email. If you have 2-step verification set up on your Google account (and you most certainly should), you'll need to complete that process as well. https://www.androidcentral.com/how-track-android-phone

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	Once you're signed in to Find My Device, you'll see a map with your current location as well as the make and model of your phone, and two options — Play Sound, and Enable Lock & Erase. Hitting the latter option will allow you to start using the Lock and Erase functions. If you've signed into more than one phone, you can select a particular device by browsing the list at the top of the screen. 1. Open **Find My Device** from your home screen or app drawer. 2. Select your phone from the **list of devices at the top of the screen**. 3. See if your phone is **discoverable**.  Find My Device Samsung Galaxy S8+ Samsung In your hand ENABLE LOCK & ERASE https://www.androidcentral.com/find-my-device

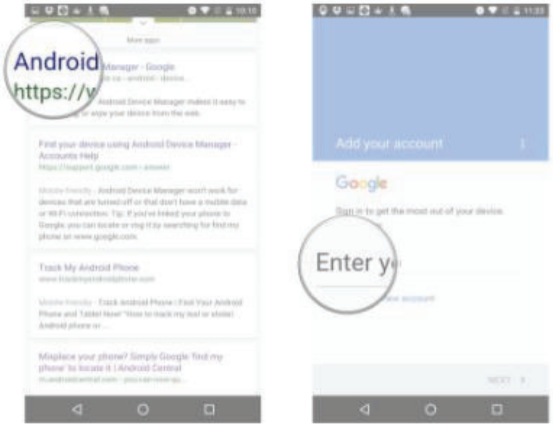
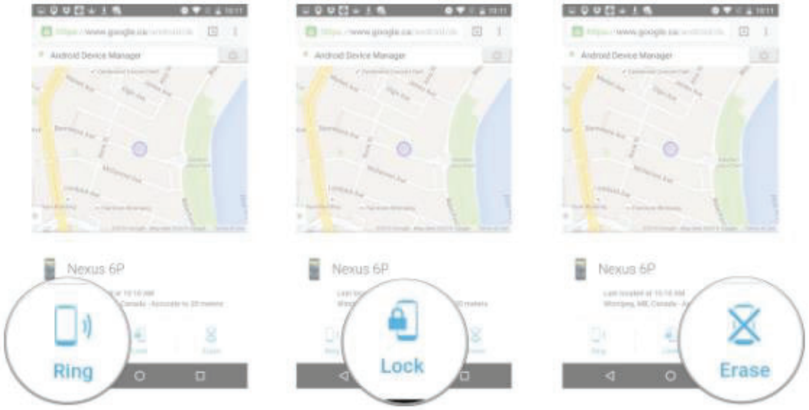
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Android https://v Add your account Google Enter y When your phone is located, you have three options to choose from: • You can **Ring** your phone so that it makes noise (even if you had it on silent). This feature is helpful if the map indicates that the phone is within earshot and you simply can't see it. • You can **Lock** your phone so that the finder can't access your home screen. This feature is most helpful if your phone wasn't previously secured with a passcode or a fingerprint sensor. • You can **Erase** your phone. This is the best option if you know for certain that you aren't likely to retrieve your phone.  https://www.androidcentral.com/how-track-android-phone

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

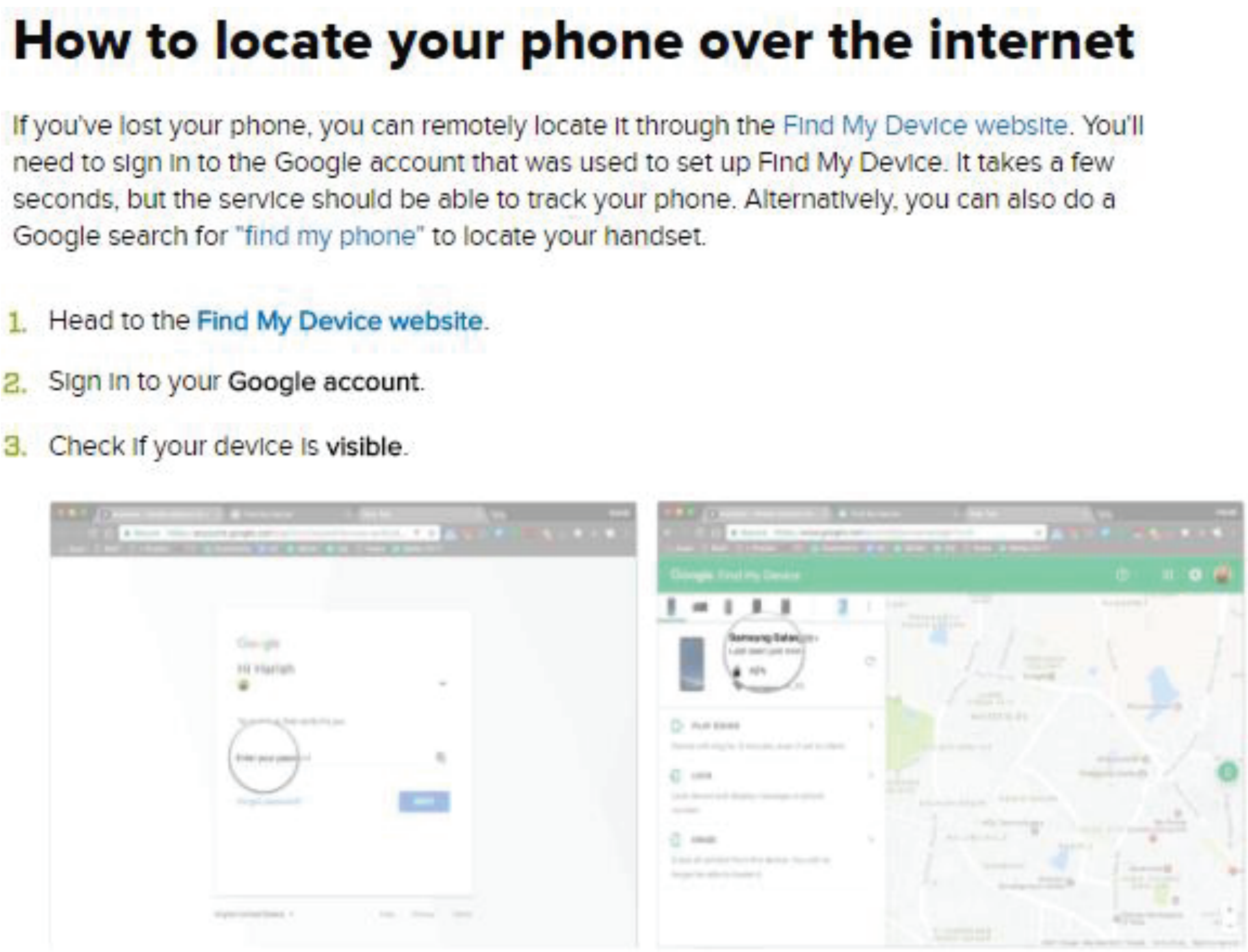
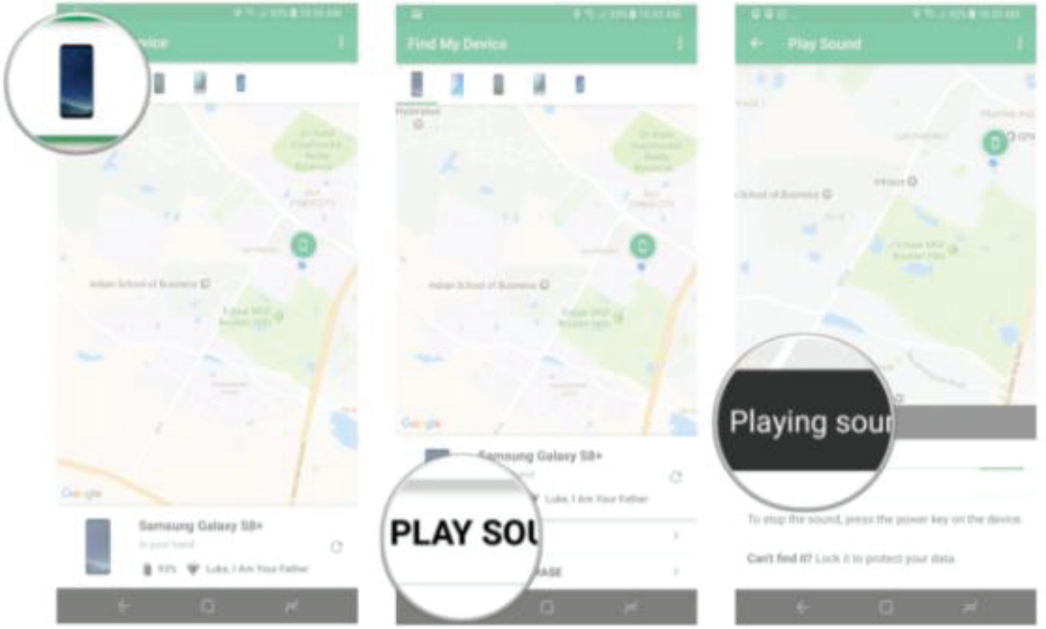
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 **How to locate your phone over the internet** If you've lost your phone, you can remotely locate it through the Find My Device website. You'll need to sign in to the Google account that was used to set up Find My Device. It takes a few seconds, but the service should be able to track your phone. Alternatively, you can also do a Google search for "find my phone" to locate your handset. 1. Head to the **Find My Device website**. 2. Sign in to your **Google account**. 3. Check if your device is **visible**. Google Hi Hannah Google Find My Device Samsung Galaxy https://www.androidcentral.com/find-my-device

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to ring your phone with Find My Device The best part about Find My Device is that it is easily accessible. If you need to locate your phone, just head to the website or log in to the service from another phone. Once you sign in to Find My Device and locate your device, you can use the **Play Sound** option, which plays a loud tone on your phone continuously at full volume for five minutes even if you turned the ringer off. Once you find your phone, you can hit the power button to stop the ringing. 1. Locate your phone on **Find My Device**. 2. Tap **Play Sound**. 3. Your device will start ringing. You can hit the power button to stop the sound.  Find My Device Play Sound Samsung Galaxy S8+ PLAY SOU Playing sour To stop the sound, press the power key on the device. Can't find it? Lock it to protect your data. https://www.androidcentral.com/find-my-device

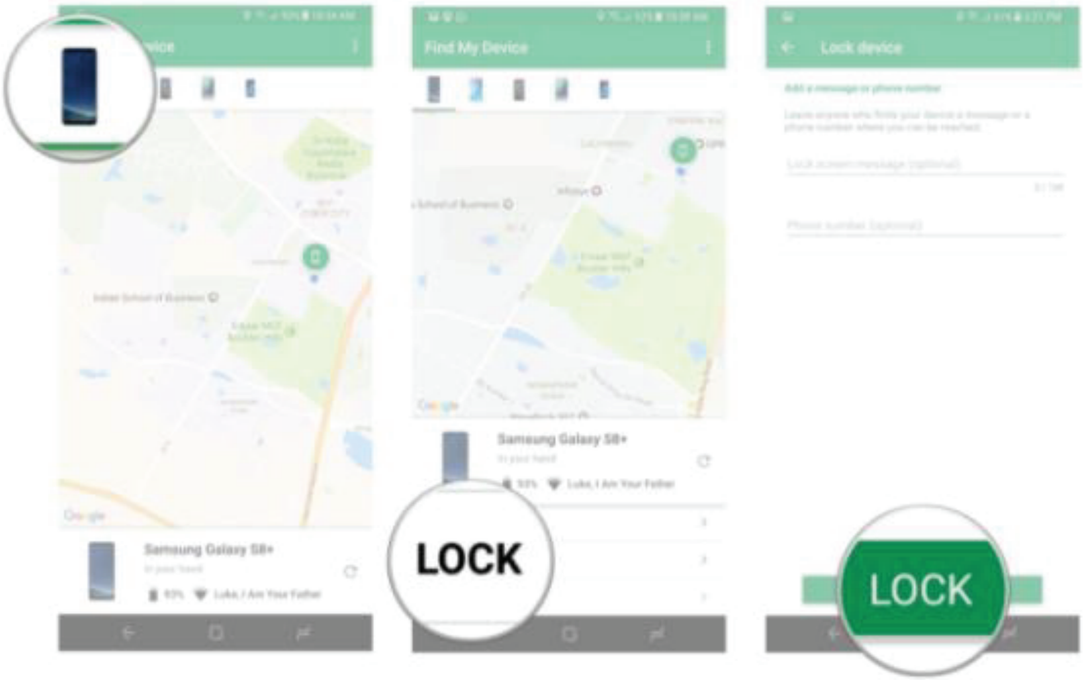
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to lock your phone with Find My Device There's also a **Lock** option that lets you set a new password to unlock the phone. You can also display a message over the lock screen and add a button to call back your number so that anyone that comes across your phone can easily get in touch with you. 1. Locate your phone on **Find My Device**. 2. Tap **Lock**. 3. Enter a **message and phone number** to display on the lock screen and tap **Lock**.  Find My Device Samsung Galaxy S8+ Luke, I Am Your Father LOCK Lock device Add a message or phone number Lock screen message (optional) Phone number (optional) LOCK https://www.androidcentral.com/find-my-device

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

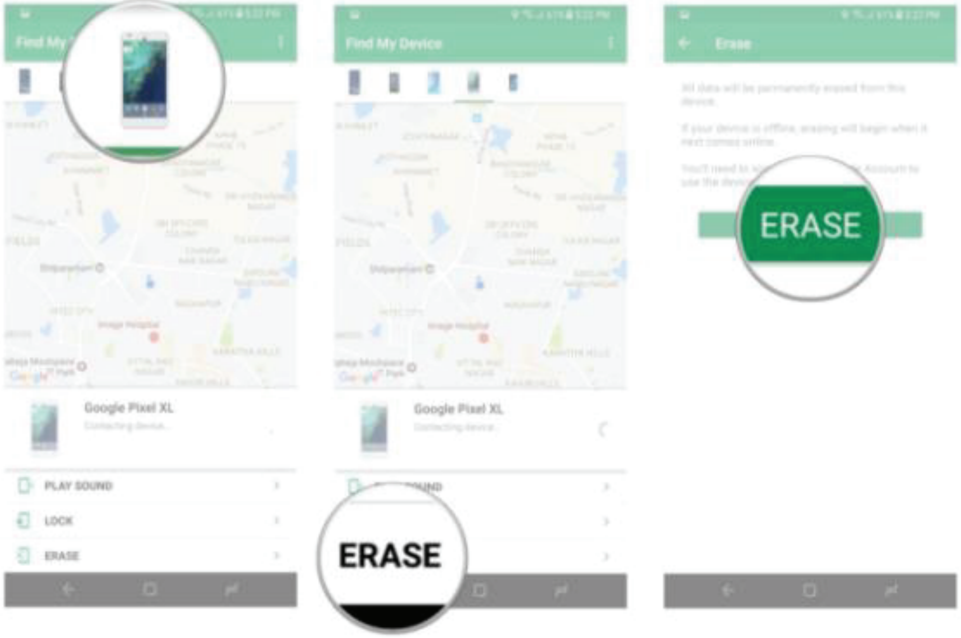
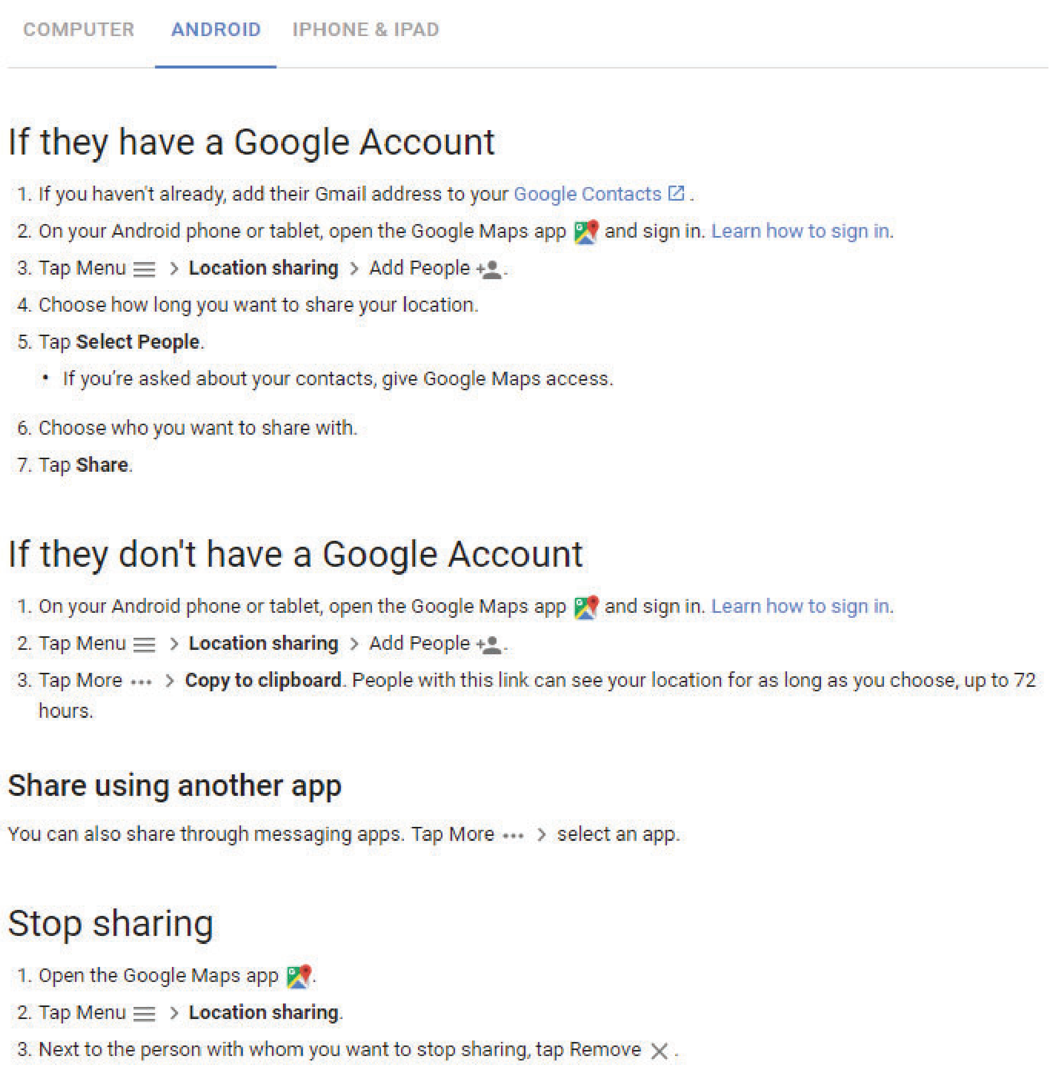
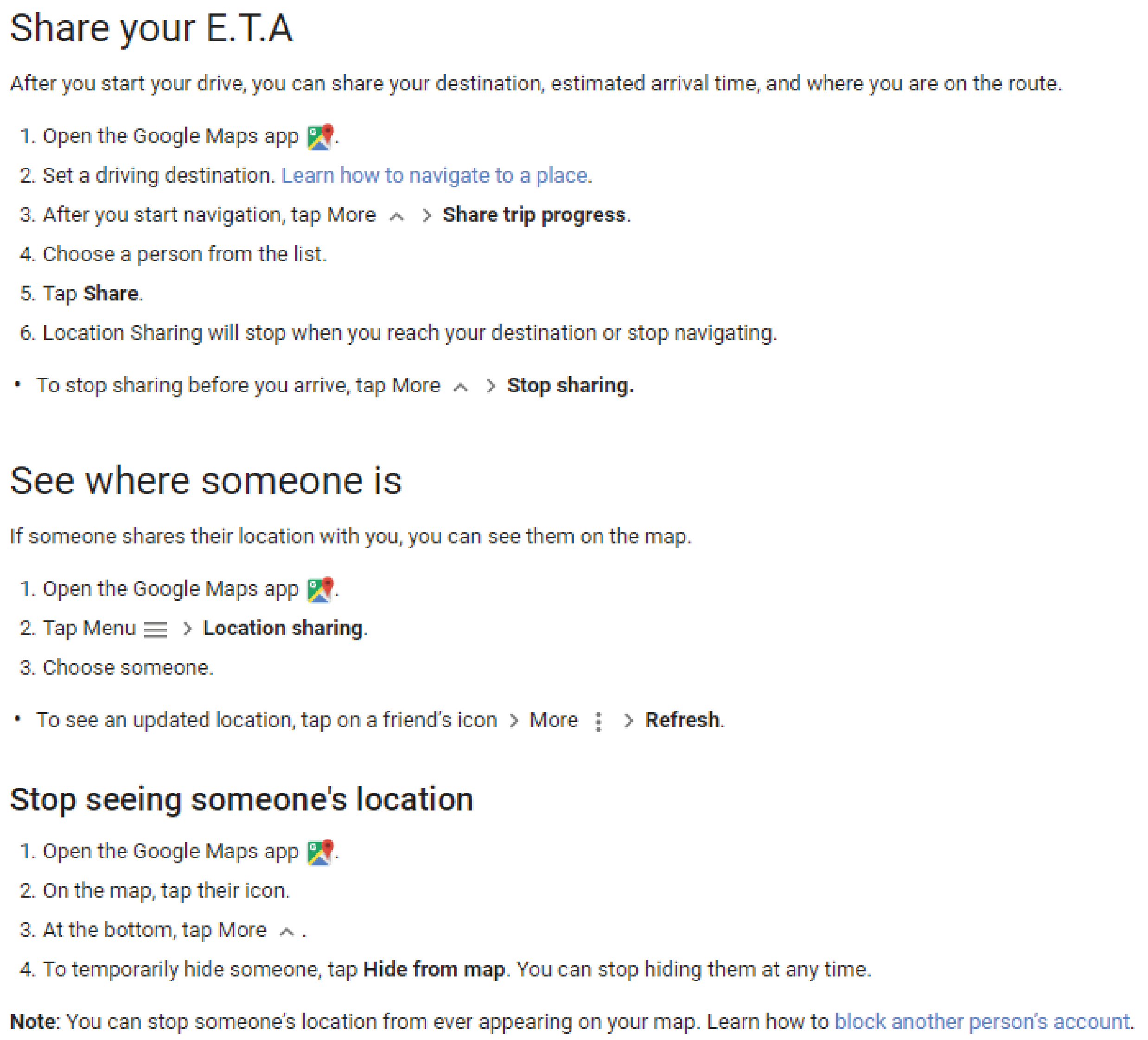
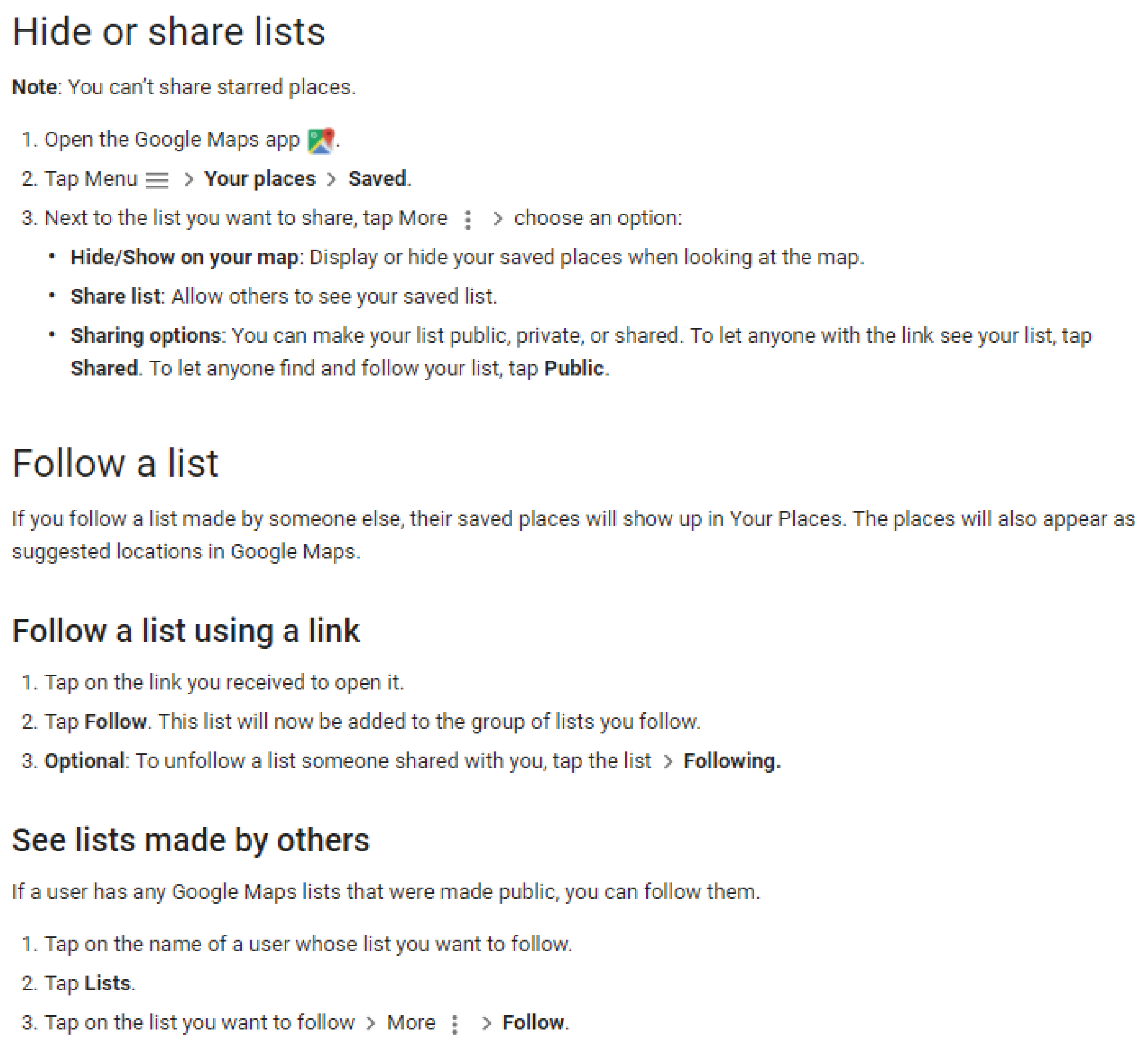
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to erase your lost phone's data remotely If you're certain that you're not going to see your phone again, there is the nuclear option of erasing the data remotely. Selecting the **Erase** option deletes all the data on your phone. The service also deletes data from a connected SD card, but there is a chance that it may not be able to, based on the manufacturer and Android platform version. Even if your phone is switched off when you send the Erase command, the factory reset process will be initiated as soon as it goes online. 1. Locate your phone on **Find My Device**. 2. Tap **Erase**. 3. Confirm deletion of data by hitting the **Erase** button.  Find My Google Pixel XL PLAY SOUND LOCK ERASE Find My Device Google Pixel XL ERASE Erase ERASE https://www.androidcentral.com/find-my-device **Exemplary Support for Google Maps**:

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 COMPUTER ANDROID IPHONE & IPAD If they have a Google Account 1. If you haven't already, add their Gmail address to your Google Contacts. 2. On your Android phone or tablet, open the Google Maps app and sign in. Learn how to sign in. 3. Tap Menu ☰ > **Location sharing** > Add People. 4. Choose how long you want to share your location. 5. Tap **Select People**. - If you're asked about your contacts, give Google Maps access. 6. Choose who you want to share with. 7. Tap **Share**. If they don't have a Google Account 1. On your Android phone or tablet, open the Google Maps app and sign in. Learn how to sign in. 2. Tap Menu ☰ > **Location sharing** > Add People. 3. Tap More ••• > **Copy to clipboard**. People with this link can see your location for as long as you choose, up to 72 hours. Share using another app You can also share through messaging apps. Tap More ••• > select an app. Stop sharing 1. Open the Google Maps app. 2. Tap Menu ☰ > **Location sharing**. 3. Next to the person with whom you want to stop sharing, tap Remove ✕. https://support.google.com/maps/answer/7326816?co=GENIE.Platform%3DAndroid&oco=1

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 ## Share your E.T.A After you start your drive, you can share your destination, estimated arrival time, and where you are on the route. 1. Open the Google Maps app . 2. Set a driving destination. Learn how to navigate to a place. 3. After you start navigation, tap More ^ > **Share trip progress**. 4. Choose a person from the list. 5. Tap **Share**. 6. Location Sharing will stop when you reach your destination or stop navigating. - To stop sharing before you arrive, tap More ^ > **Stop sharing.** ## See where someone is If someone shares their location with you, you can see them on the map. 1. Open the Google Maps app . 2. Tap Menu ≡ > **Location sharing**. 3. Choose someone. - To see an updated location, tap on a friend's icon > More ⋮ > **Refresh**. ## Stop seeing someone's location 1. Open the Google Maps app . 2. On the map, tap their icon. 3. At the bottom, tap More ^ . 4. To temporarily hide someone, tap **Hide from map**. You can stop hiding them at any time. **Note**: You can stop someone's location from ever appearing on your map. Learn how to block another person's account. https://support.google.com/maps/answer/7326816?co=GENIE.Platform%3DAndroid&oco=1

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

Hide or share lists

Note: You can't share starred places.

1. Open the Google Maps app 🗺.
2. Tap Menu ☰ > **Your places** > **Saved**.
3. Next to the list you want to share, tap More ⋮ > choose an option:
   - **Hide/Show on your map**: Display or hide your saved places when looking at the map.
   - **Share list**: Allow others to see your saved list.
   - **Sharing options**: You can make your list public, private, or shared. To let anyone with the link see your list, tap **Shared**. To let anyone find and follow your list, tap **Public**.

Follow a list

If you follow a list made by someone else, their saved places will show up in Your Places. The places will also appear as suggested locations in Google Maps.

Follow a list using a link

1. Tap on the link you received to open it.
2. Tap **Follow**. This list will now be added to the group of lists you follow.
3. **Optional**: To unfollow a list someone shared with you, tap the list > **Following.**

See lists made by others

If a user has any Google Maps lists that were made public, you can follow them.

1. Tap on the name of a user whose list you want to follow.
2. Tap **Lists**.
3. Tap on the list you want to follow > More ⋮ > **Follow**.

https://support.google.com/maps/answer/7280933?hl=en&ref_topic=7301134&co=GENIE.Platform%3DA

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

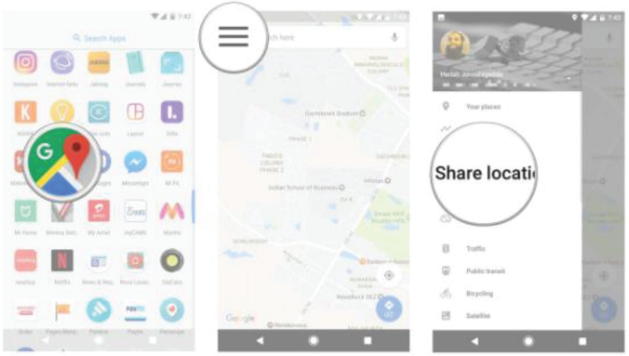
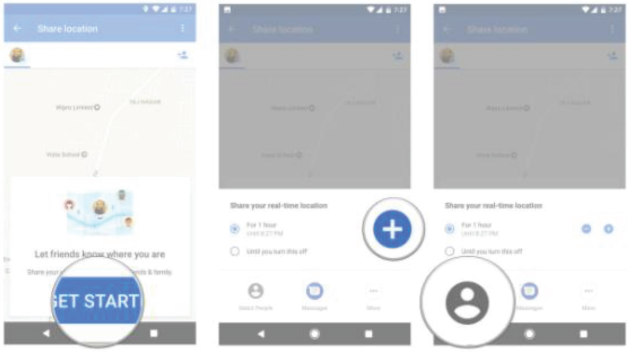
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	ndroid&oco=1 **How to share your location in Google Maps** 1. Open **Google Maps** from the app drawer or the home screen. 2. Tap the **hamburger menu** (the three horizontal lines) on the top left corner of the screen. 3. Select **Share location**.  4. Tap **Get Started**. 5. Use the **+** icon to select a time period or select the **Until you turn this off** setting to share your location indefinitely. 6. Tap **Select People**.  Share location Let friends know where you are ET START Share your real-time location For 1 hour Until you turn this off https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

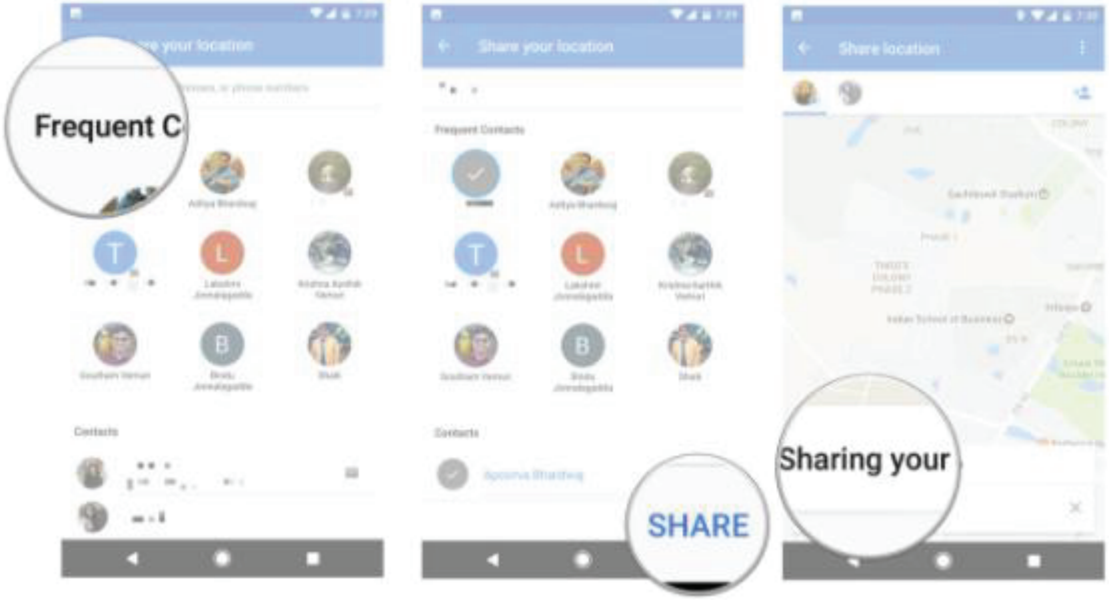
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	7. You'll see a list of your frequent contacts at the top, along with a full list of contacts. **Pick the contacts** by tapping their name. 8. Once you've selected the contacts you want to share your location to, tap **Share**. 9. You'll see a message saying that the selected contact can view your location.  Frequent C Share your location Frequent Contacts Contacts SHARE Share location Sharing your https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

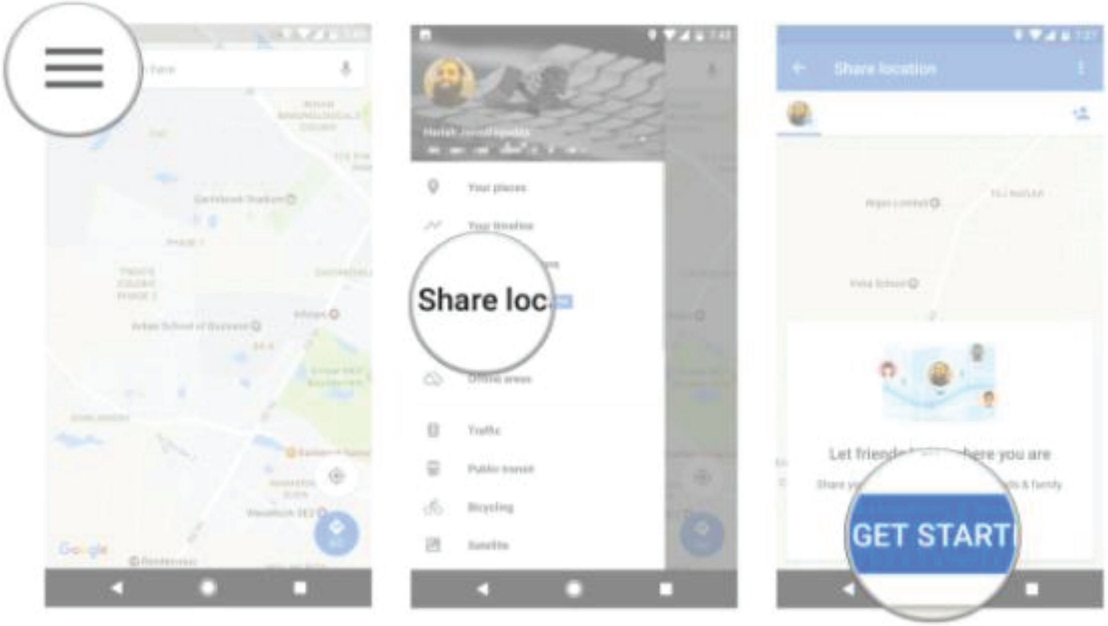
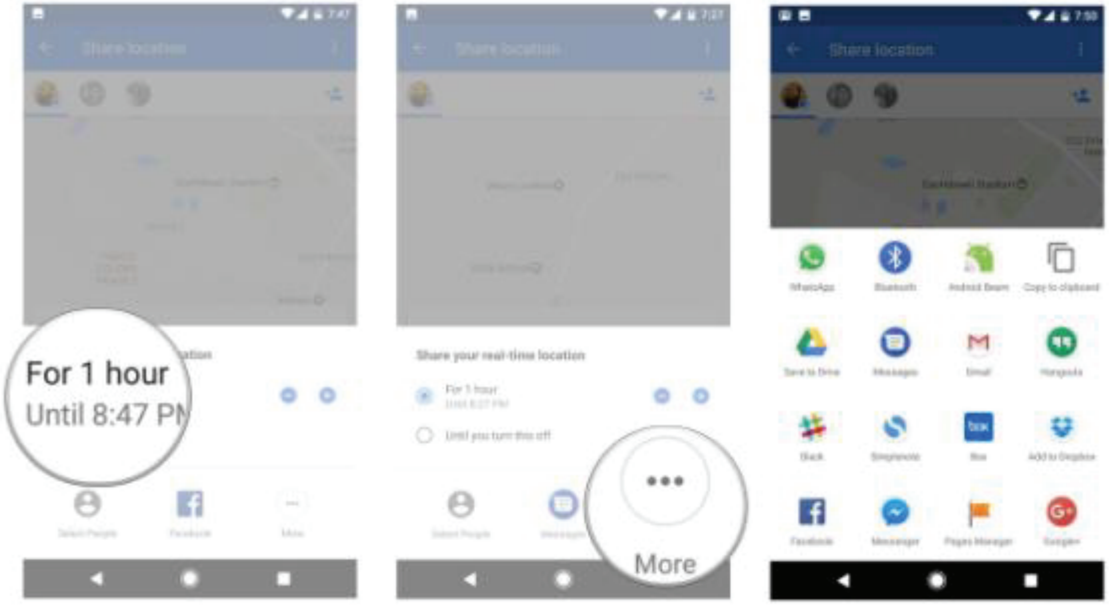
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to create a shareable link You can also create a link and use it to share your location easily. Here's how to do it: 1. Tap the **hamburger menu** on the top left corner of the screen. 2. Select **Share location**. 3. Tap **Get Started**.  Share loc Share location Let friends ... where you are GET START https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	4. Select the **amount of time** you want to share your location. 5. Tap **More**. 6. Select **your app of choice** to create and send a unique URL that broadcasts your current location. You can email it, send the link via Messenger, or even tweet it to the intended recipient.  Share location For 1 hour Until 8:47 PM Share your real-time location For 1 hour Until you turn this off More WhatsApp Bluetooth Android Beam Copy to clipboard Save to Drive Messages Email Hangouts Slack Simplenote Box Add to Dropbox Facebook Messenger Pages Manager Google+ https://www.androidcentral.com/how-share-location-google-maps

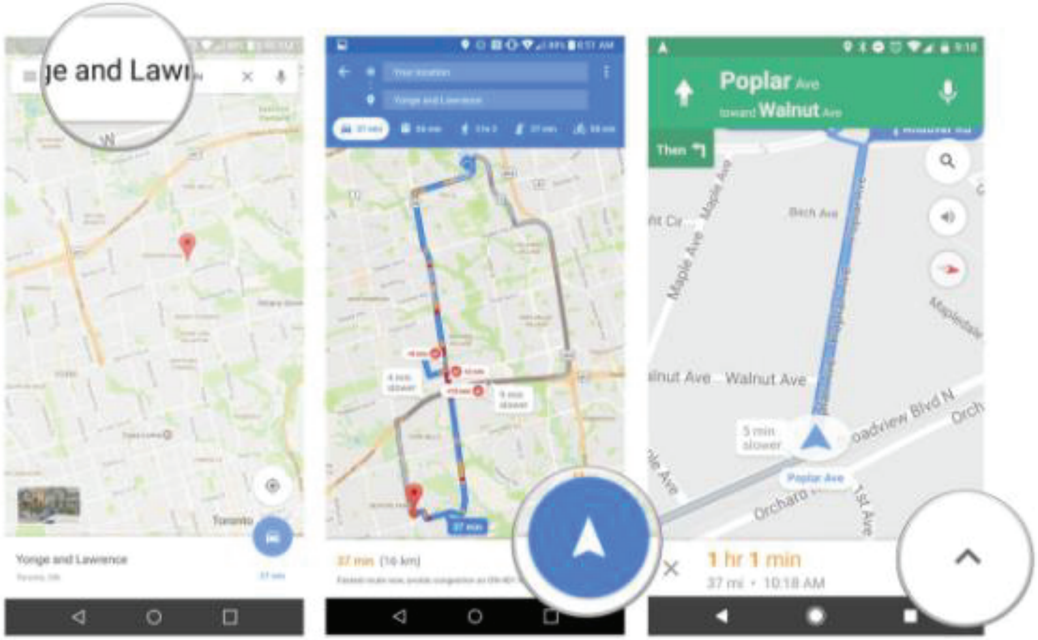
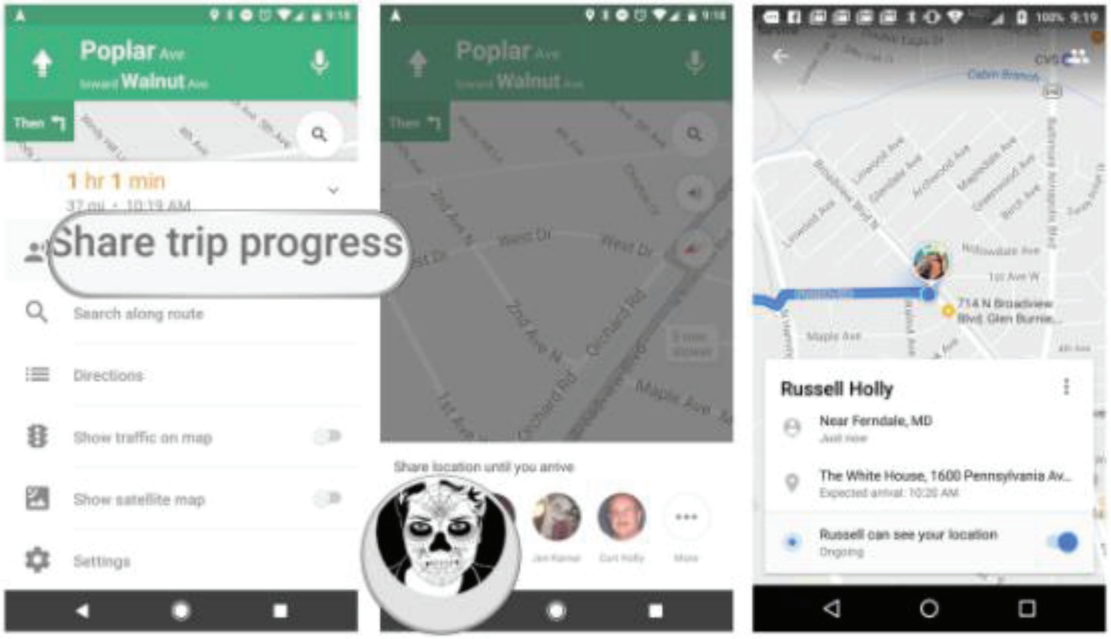
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	How to share your navigation directions while you walk, drive or transit One of the best ways to share your location with someone is during a drive, walk or transit. If you're meeting a friend or family member somewhere, or navigating towards their home, Google Maps lets you share your location with them for the duration of the trip. It's magic! 1. In the **search bar** enter your destination. 2. Pick your navigation type (drive, transit, walk) and press the blue **navigate button**. 3. Tap the arrow next to the **time-to-destination** number at the bottom of the screen.  ge and Lawr Yonge and Lawrence Toronto 37 min (16 km) 37 min Poplar Ave toward Walnut Ave Then Birch Ave Maple Ave Walnut Ave 5 min slower Broadview Blvd N Orchard 1st Ave Poplar Ave 1 hr 1 min 37 mi • 10:18 AM https://www.androidcentral.com/how-share-location-google-maps

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	4. Tap Share trip progress. 5. Choose one or more contacts to share trip progress.  Poplar Ave toward Walnut Ave Then 1 hr 1 min 37 mi · 10:19 AM Share trip progress Search along route Directions Show traffic on map Show satellite map Settings Share location until you arrive Jen Karner Carl Holly More Russell Holly Near Ferndale, MD Just now The White House, 1600 Pennsylvania Av... Expected arrival: 10:20 AM Russell can see your location Ongoing 714 N Broadview Blvd, Glen Burnie... Baltimore Annapolis Blvd Cabin Branch CVS You can also stop sharing your location with someone before a trip ends. https://www.androidcentral.com/how-share-location-google-maps

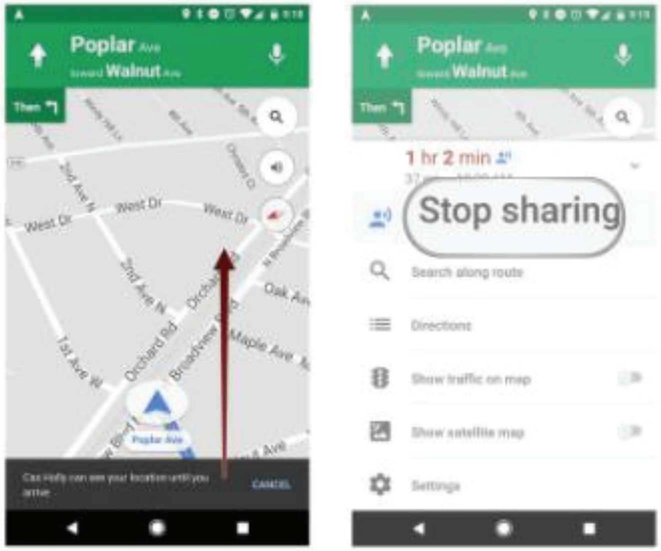
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	1. Tap the arrow next to the **time-to-destination** number at the bottom of the screen. 2. Tap **Stop sharing.**  Poplar Ave toward Walnut Ave Then CANCEL 1 hr 2 min Stop sharing Search along route Directions Show traffic on map Show satellite map Settings That's it! Are you excited that location sharing is back in Google Maps? How often do you use the feature? https://www.androidcentral.com/how-share-location-google-maps As shown below, a group may also be defined within Google Contacts.

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

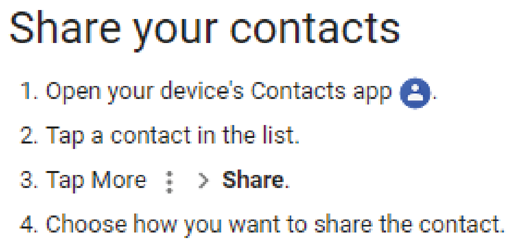
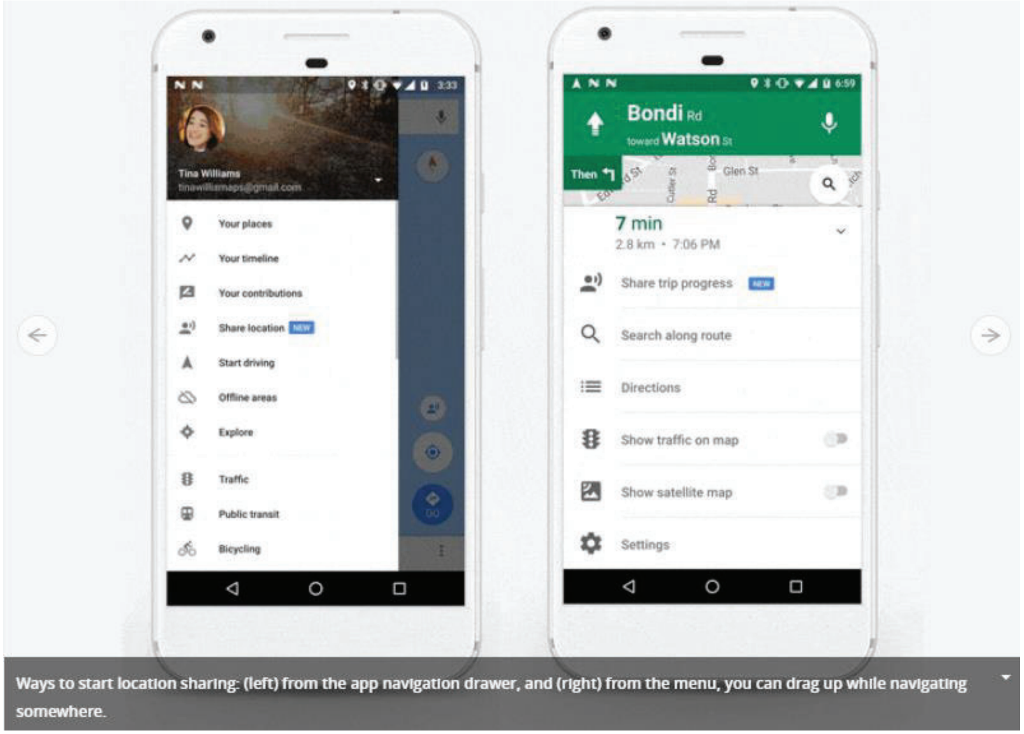
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Share your contacts 1. Open your device's Contacts app. 2. Tap a contact in the list. 3. Tap More ⋮ > **Share**. 4. Choose how you want to share the contact. https://support.google.com/android/answer/6118731?hl=en&ref_topic=6118711  Tina Williams tinawilliamsmaps@gmail.com Your places Your timeline Your contributions Share location NEW Start driving Offline areas Explore Traffic Public transit Bicycling Bondi Rd toward Watson St Then Glen St 7 min 2.8 km · 7:06 PM Share trip progress NEW Search along route Directions Show traffic on map Show satellite map Settings Ways to start location sharing: (left) from the app navigation drawer, and (right) from the menu, you can drag up while navigating somewhere. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

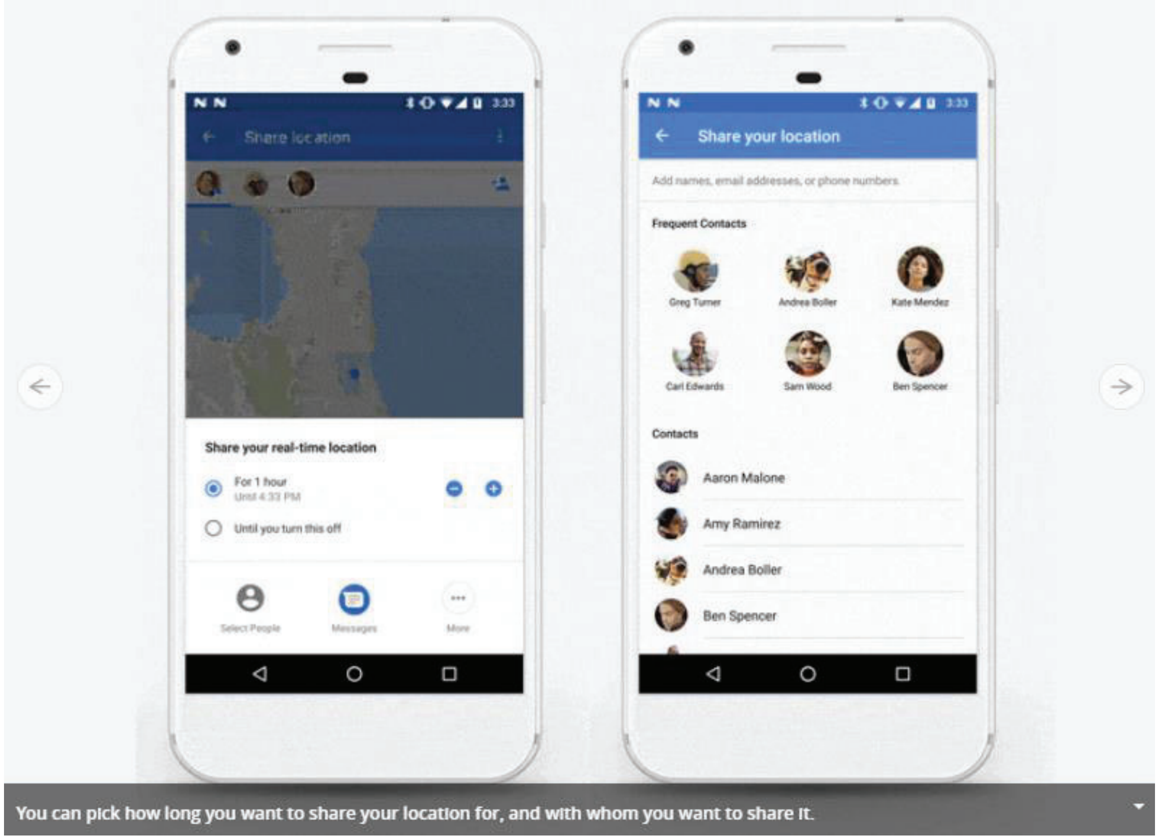
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Share location Share your real-time location ● For 1 hour Until 4:33 PM ○ Until you turn this off Select People · Messages · More Share your location Add names, email addresses, or phone numbers Frequent Contacts: Greg Turner, Andrea Boller, Kate Mendez, Carl Edwards, Sam Wood, Ben Spencer Contacts: Aaron Malone, Amy Ramirez, Andrea Boller, Ben Spencer You can pick how long you want to share your location for, and with whom you want to share it. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

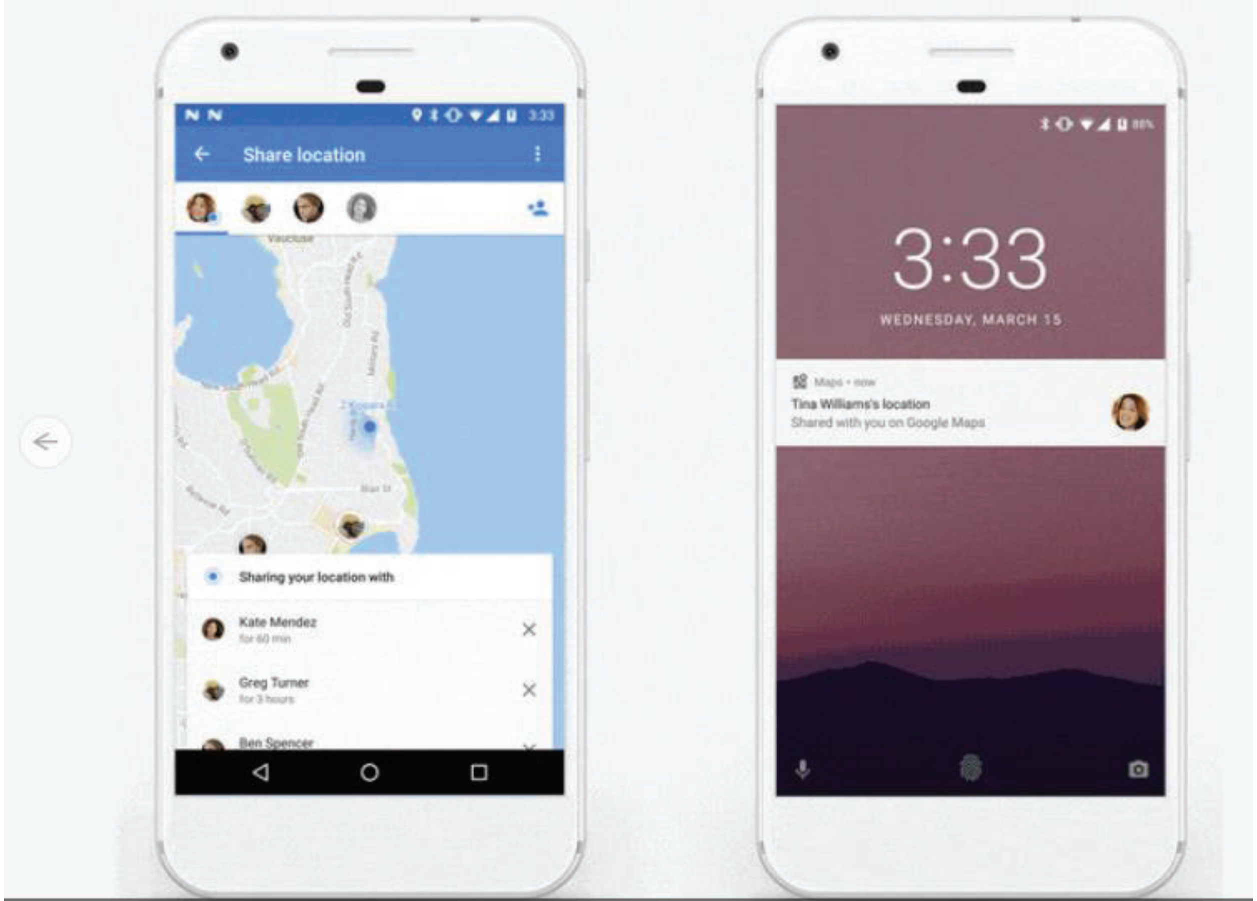
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Share location Sharing your location with Kate Mendez for 60 min Greg Turner for 3 hours Ben Spencer 3:33 WEDNESDAY, MARCH 15 Maps • now Tina Williams's location Shared with you on Google Maps People you've shared with will get a notification from Google Maps. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

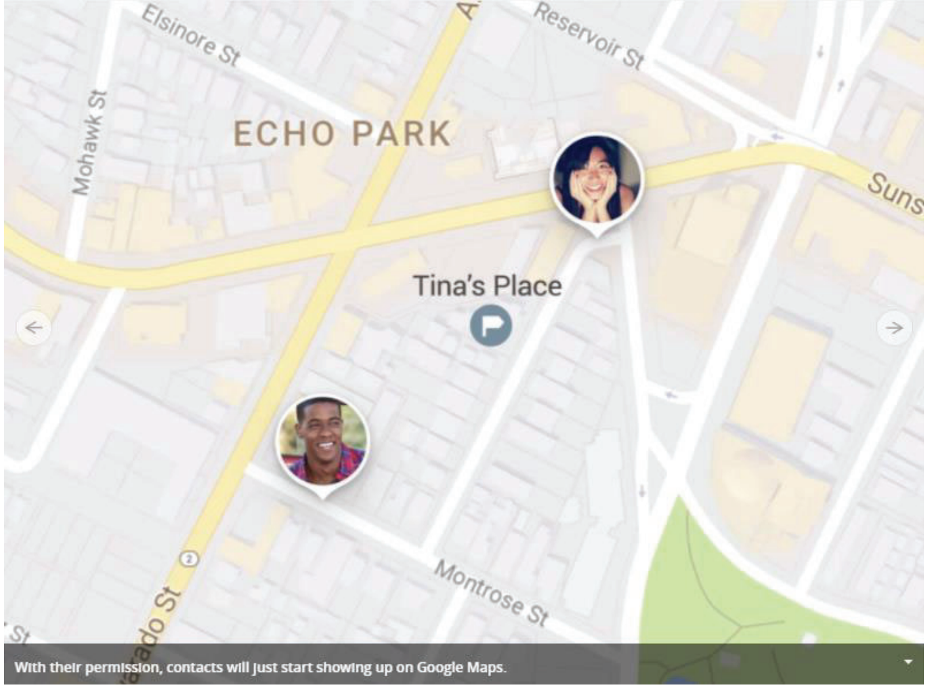
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 Elsinore St Reservoir St Mohawk St ECHO PARK Suns Tina's Place Montrose St ado St With their permission, contacts will just start showing up on Google Maps. https://arstechnica.com/gadgets/2017/03/location-sharing-finally-returns-to-google-maps/ **<u>Exemplary Find My Device Screenshots</u>**:

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

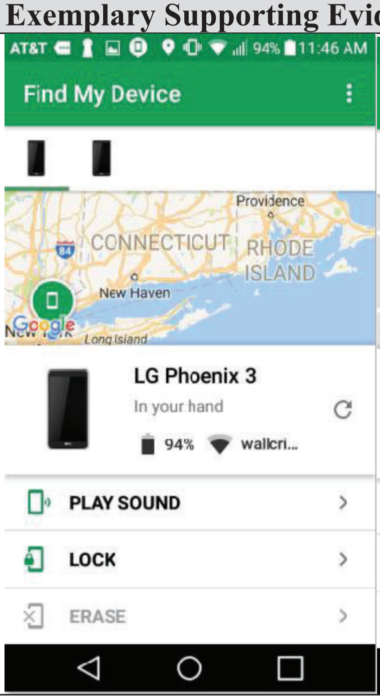
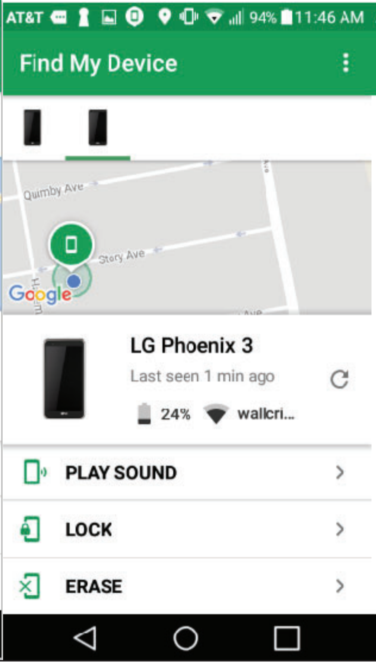
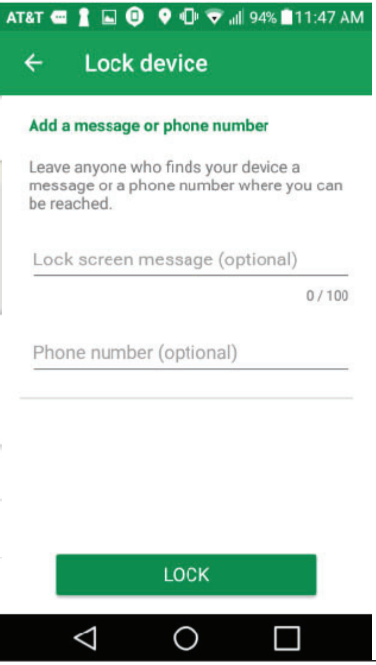
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 94% 11:46 AM Find My Device LG Phoenix 3 In your hand 94% wallcri... PLAY SOUND LOCK ERASE  AT&T 94% 11:46 AM Find My Device LG Phoenix 3 Last seen 1 min ago 24% wallcri... PLAY SOUND LOCK ERASE  AT&T 94% 11:47 AM Lock device Add a message or phone number Leave anyone who finds your device a message or a phone number where you can be reached. Lock screen message (optional) 0 / 100 Phone number (optional) LOCK

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

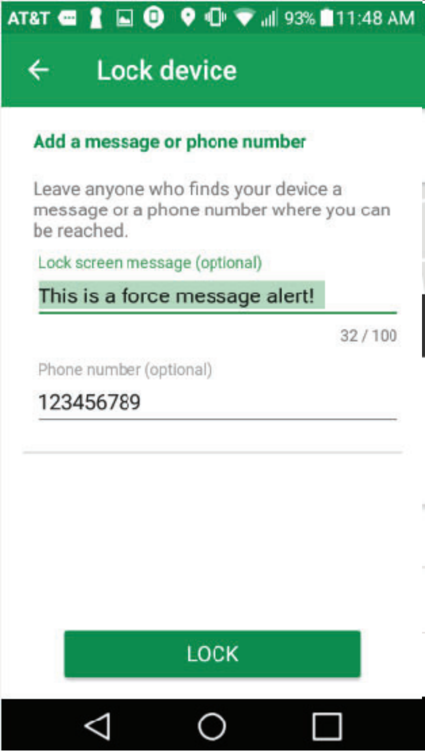
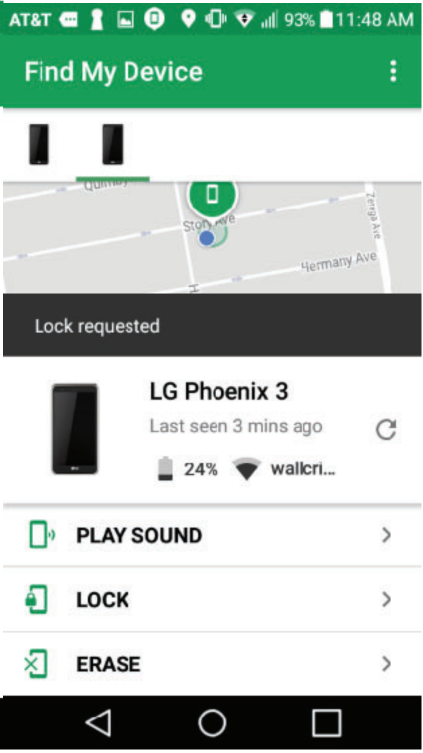
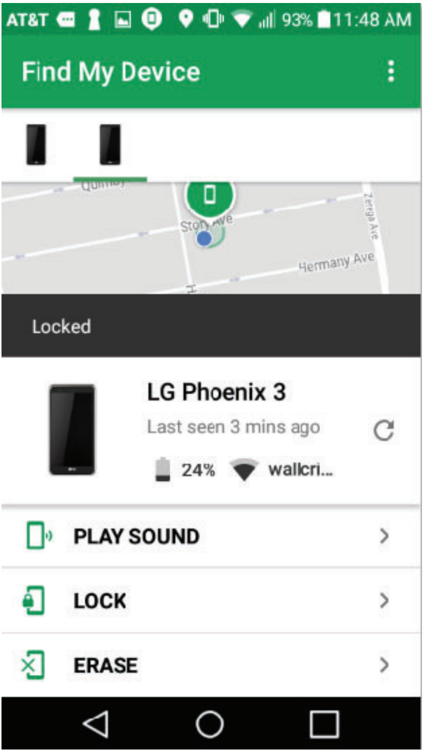
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 93% 11:48 AM Lock device Add a message or phone number Leave anyone who finds your device a message or a phone number where you can be reached. Lock screen message (optional) This is a force message alert! 32 / 100 Phone number (optional) 123456789 LOCK  AT&T 93% 11:48 AM Find My Device Lock requested LG Phoenix 3 Last seen 3 mins ago 24% wallcri... PLAY SOUND LOCK ERASE  AT&T 93% 11:48 AM Find My Device Locked LG Phoenix 3 Last seen 3 mins ago 24% wallcri... PLAY SOUND LOCK ERASE

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

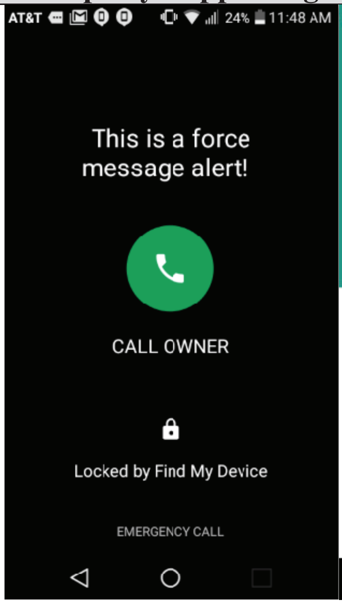
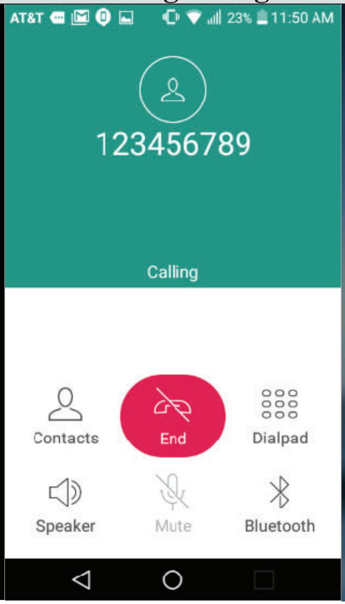
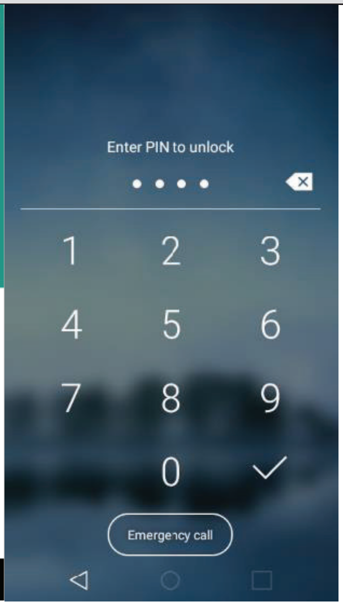
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 24% 11:48 AM This is a force message alert! CALL OWNER Locked by Find My Device EMERGENCY CALL  AT&T 23% 11:50 AM 123456789 Calling Contacts End Dialpad Speaker Mute Bluetooth  Enter PIN to unlock 1 2 3 4 5 6 7 8 9 0 Emergency call **<u>Exemplary Find My Device Screenshots</u>**:

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

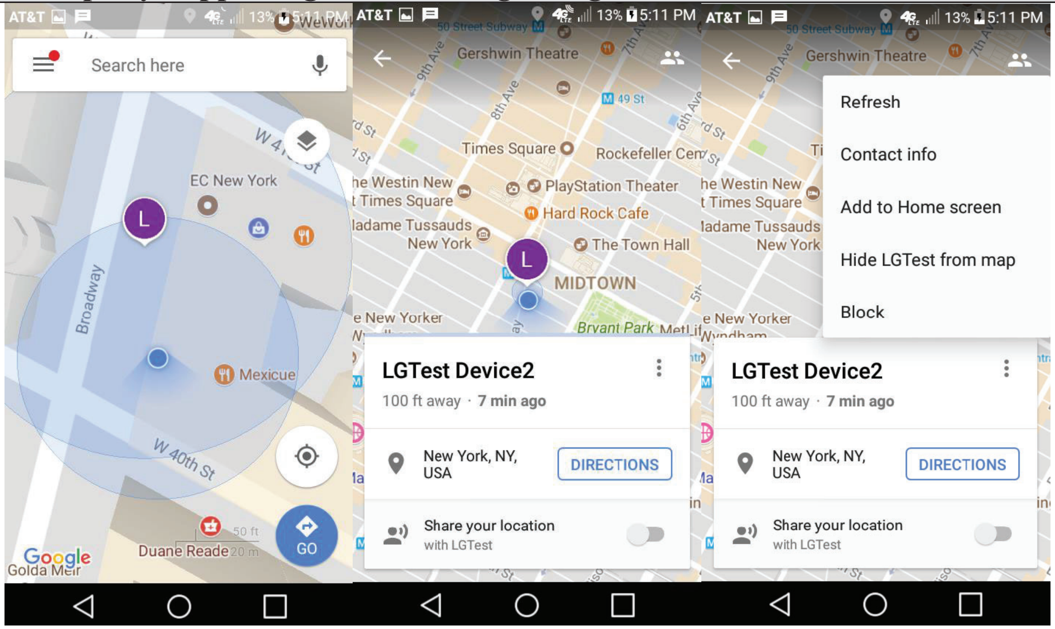
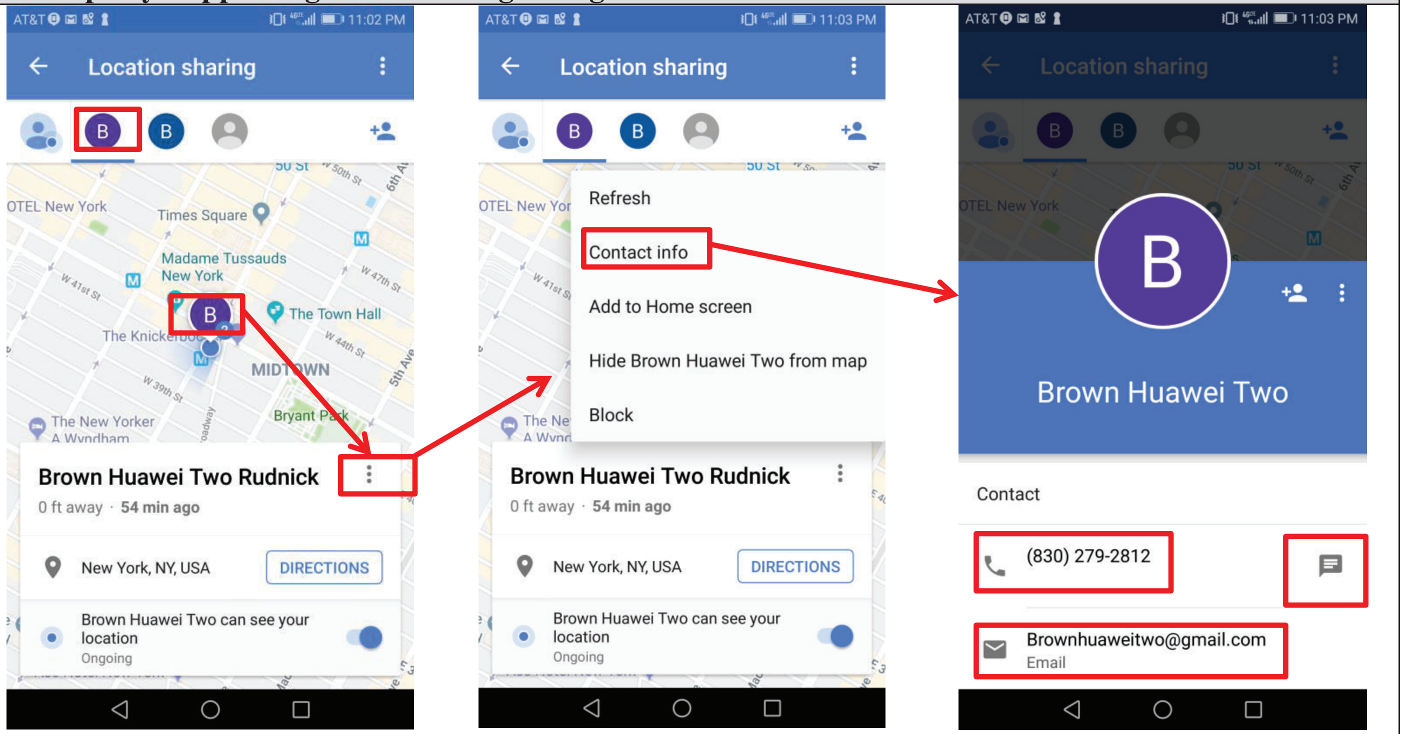
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 AT&T 13% 5:11 PM Search here EC New York Broadway Mexicue W 40th St Duane Reade GO Google Gershwin Theatre Times Square Rockefeller Cen PlayStation Theater Hard Rock Cafe The Town Hall MIDTOWN Bryant Park Refresh Contact info Add to Home screen Hide LGTest from map Block LGTest Device2 100 ft away · 7 min ago New York, NY, USA DIRECTIONS Share your location with LGTest Upon information and belief, selection of contact info results in the presentation of actions to be chosen. See, e.g.,

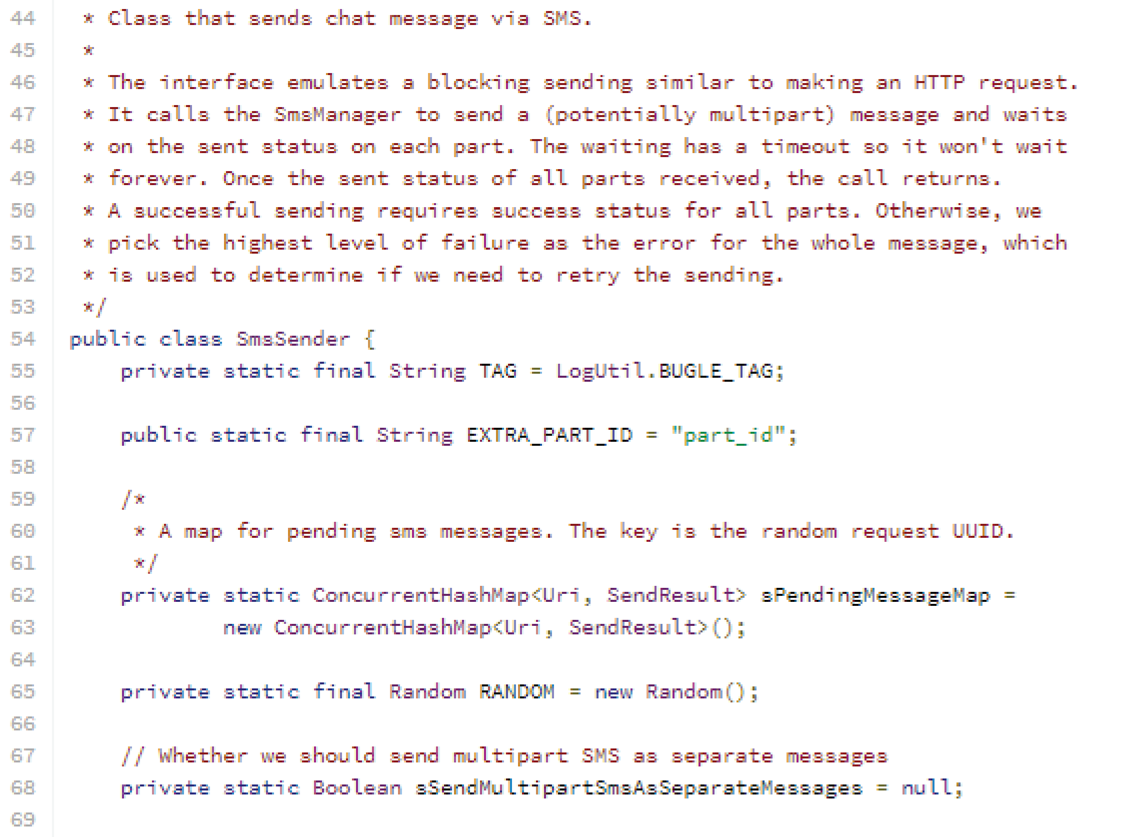
Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products



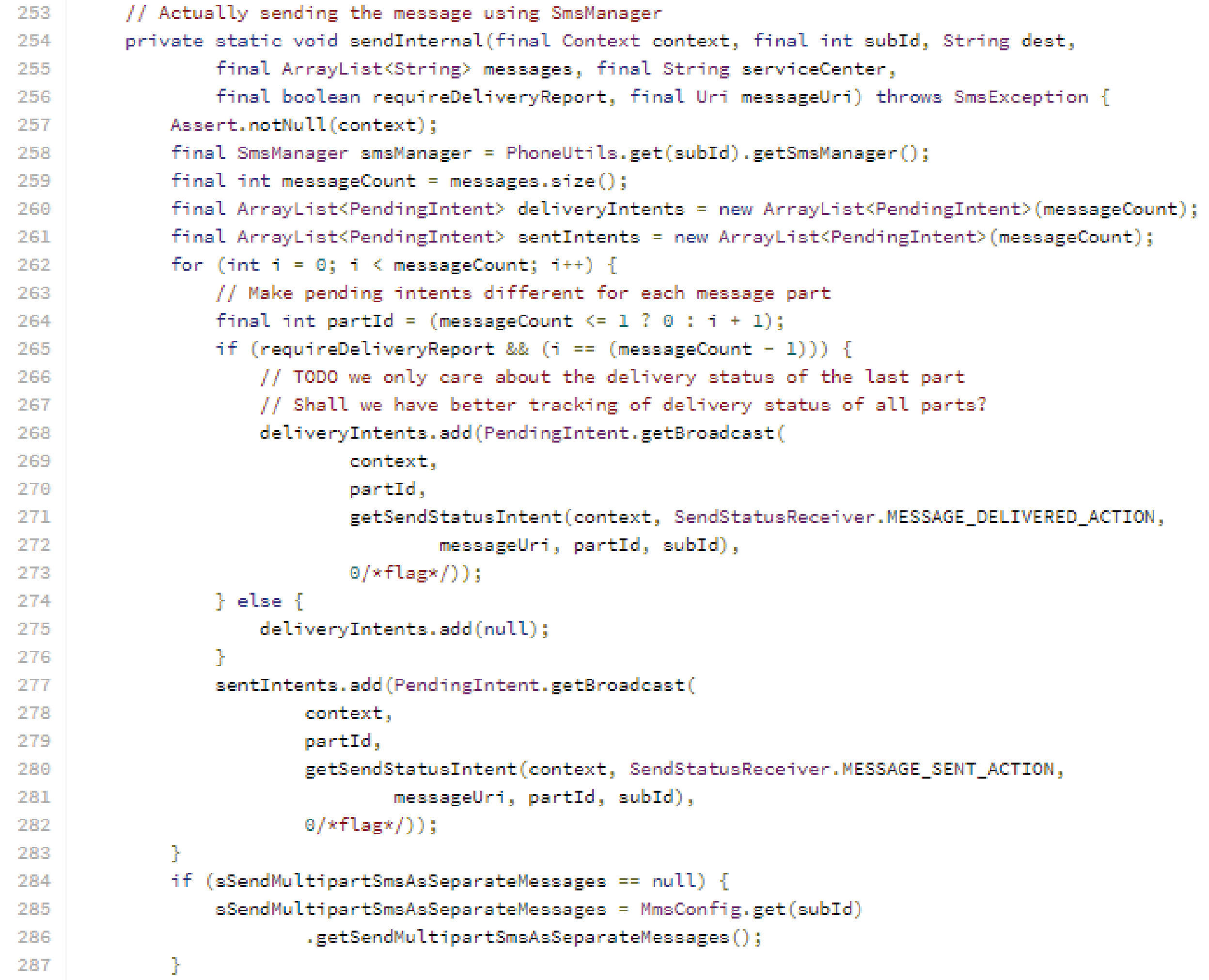
Exemplary Source Code:

The above functionality is performed at least in part by the following publicly available source code and/or source code that invokes or is invoked by the following source code (or a substantially similar copy compiled and loaded onto the Accused Products by ZTE). AGIS reserves the right to supplement these contentions with additional source code as discovery progresses and as additional source code is made available. AGIS reserves the right to supplement these contentions with additional source code as discovery progresses and as additional source code is made available.

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/SmsSender.java

```java
 * Class that sends chat message via SMS.
 *
 * The interface emulates a blocking sending similar to making an HTTP request.
 * It calls the SmsManager to send a (potentially multipart) message and waits
 * on the sent status on each part. The waiting has a timeout so it won't wait
 * forever. Once the sent status of all parts received, the call returns.
 * A successful sending requires success status for all parts. Otherwise, we
 * pick the highest level of failure as the error for the whole message, which
 * is used to determine if we need to retry the sending.
 */
public class SmsSender {
    private static final String TAG = LogUtil.BUGLE_TAG;

    public static final String EXTRA_PART_ID = "part_id";

    /*
     * A map for pending sms messages. The key is the random request UUID.
     */
    private static ConcurrentHashMap<Uri, SendResult> sPendingMessageMap =
            new ConcurrentHashMap<Uri, SendResult>();

    private static final Random RANDOM = new Random();

    // Whether we should send multipart SMS as separate messages
    private static Boolean sSendMultipartSmsAsSeparateMessages = null;

```

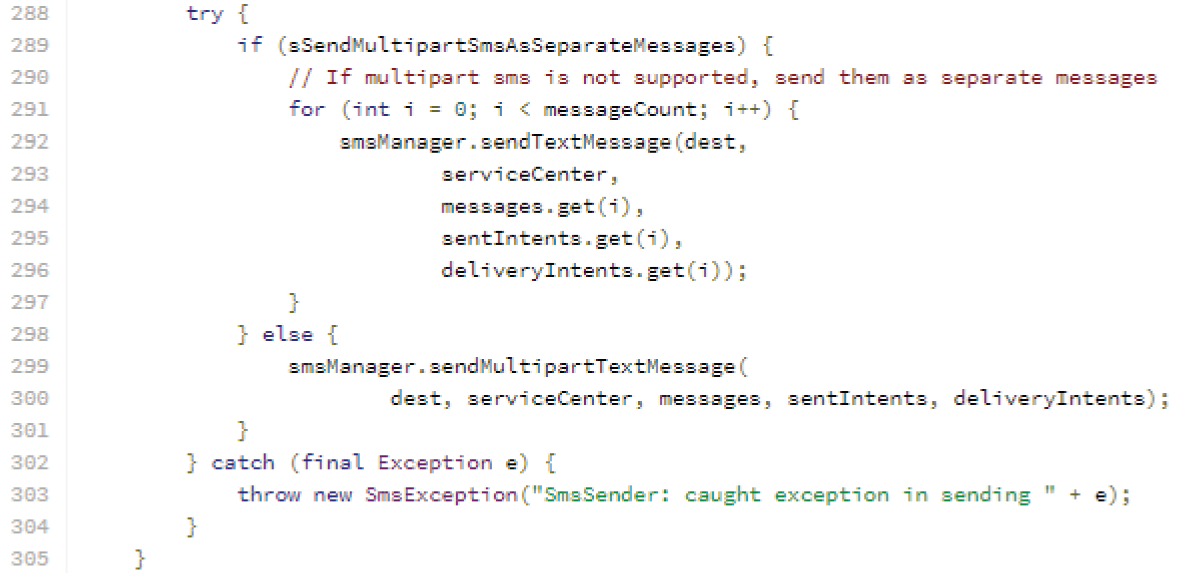
Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
    // Actually sending the message using SmsManager
    private static void sendInternal(final Context context, final int subId, String dest,
            final ArrayList<String> messages, final String serviceCenter,
            final boolean requireDeliveryReport, final Uri messageUri) throws SmsException {
        Assert.notNull(context);
        final SmsManager smsManager = PhoneUtils.get(subId).getSmsManager();
        final int messageCount = messages.size();
        final ArrayList<PendingIntent> deliveryIntents = new ArrayList<PendingIntent>(messageCount);
        final ArrayList<PendingIntent> sentIntents = new ArrayList<PendingIntent>(messageCount);
        for (int i = 0; i < messageCount; i++) {
            // Make pending intents different for each message part
            final int partId = (messageCount <= 1 ? 0 : i + 1);
            if (requireDeliveryReport && (i == (messageCount - 1))) {
                // TODO we only care about the delivery status of the last part
                // Shall we have better tracking of delivery status of all parts?
                deliveryIntents.add(PendingIntent.getBroadcast(
                        context,
                        partId,
                        getSendStatusIntent(context, SendStatusReceiver.MESSAGE_DELIVERED_ACTION,
                                messageUri, partId, subId),
                        0/*flag*/));
            } else {
                deliveryIntents.add(null);
            }
            sentIntents.add(PendingIntent.getBroadcast(
                    context,
                    partId,
                    getSendStatusIntent(context, SendStatusReceiver.MESSAGE_SENT_ACTION,
                            messageUri, partId, subId),
                    0/*flag*/));
        }
        if (sSendMultipartSmsAsSeparateMessages == null) {
            sSendMultipartSmsAsSeparateMessages = MmsConfig.get(subId)
                    .getSendMultipartSmsAsSeparateMessages();
        }
```

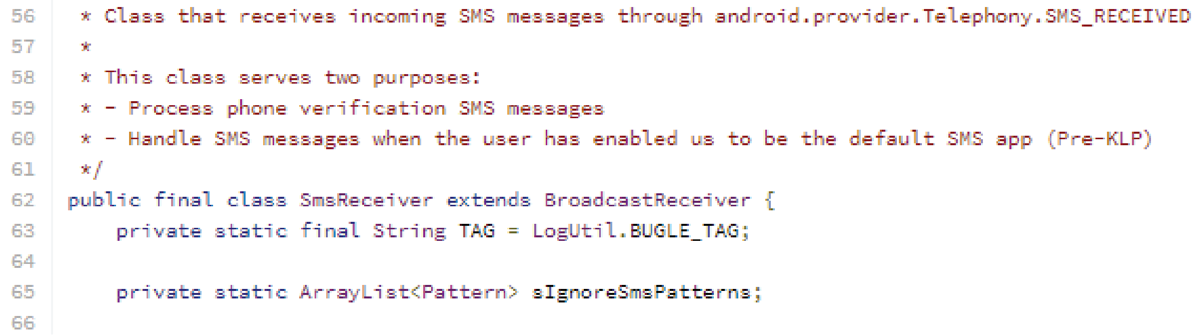
https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/SmsSender.java

US9467838	Exemplary Supporting Evidence Regarding Accused Products



```
        try {
            if (sSendMultipartSmsAsSeparateMessages) {
                // If multipart sms is not supported, send them as separate messages
                for (int i = 0; i < messageCount; i++) {
                    smsManager.sendTextMessage(dest,
                            serviceCenter,
                            messages.get(i),
                            sentIntents.get(i),
                            deliveryIntents.get(i));
                }
            } else {
                smsManager.sendMultipartTextMessage(
                        dest, serviceCenter, messages, sentIntents, deliveryIntents);
            }
        } catch (final Exception e) {
            throw new SmsException("SmsSender: caught exception in sending " + e);
        }
    }
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/SmsSender.java

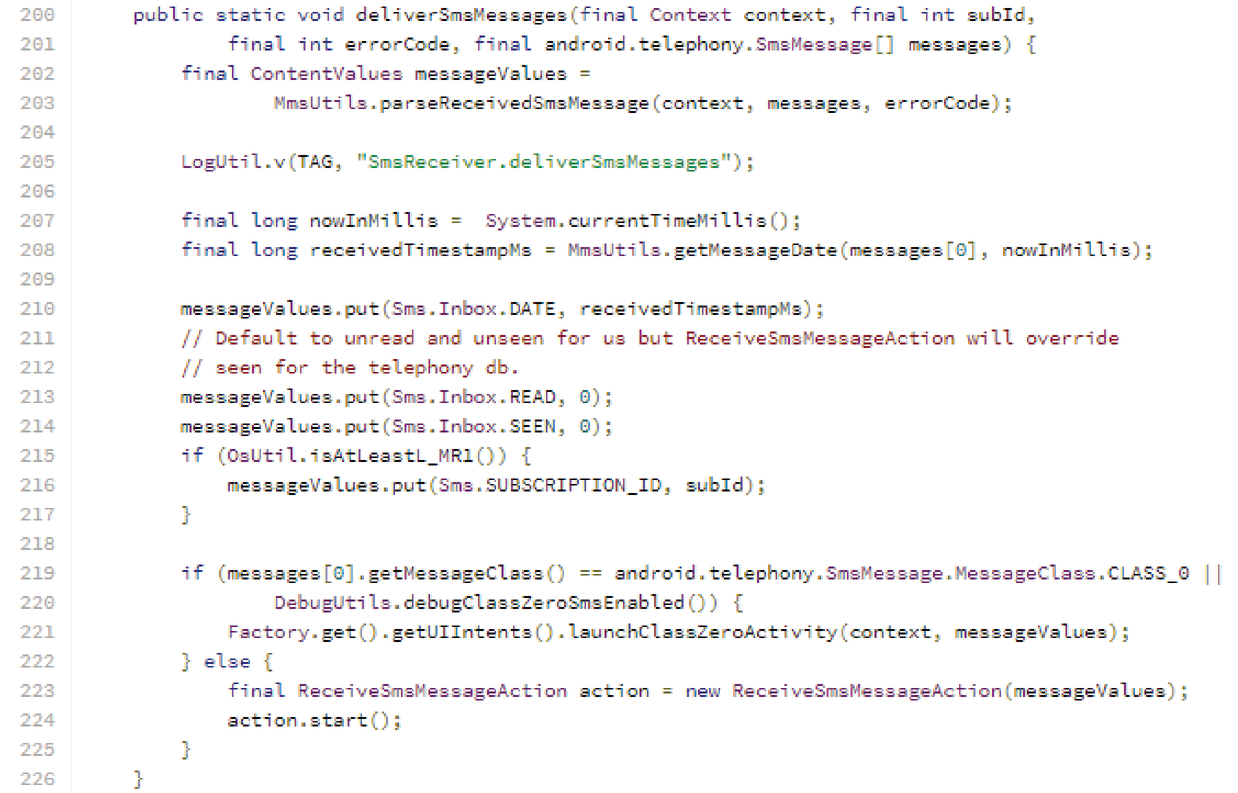


```
 * Class that receives incoming SMS messages through android.provider.Telephony.SMS_RECEIVED
 *
 * This class serves two purposes:
 * - Process phone verification SMS messages
 * - Handle SMS messages when the user has enabled us to be the default SMS app (Pre-KLP)
 */
public final class SmsReceiver extends BroadcastReceiver {
    private static final String TAG = LogUtil.BUGLE_TAG;

    private static ArrayList<Pattern> sIgnoreSmsPatterns;

```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/receiver/SmsReceiver.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

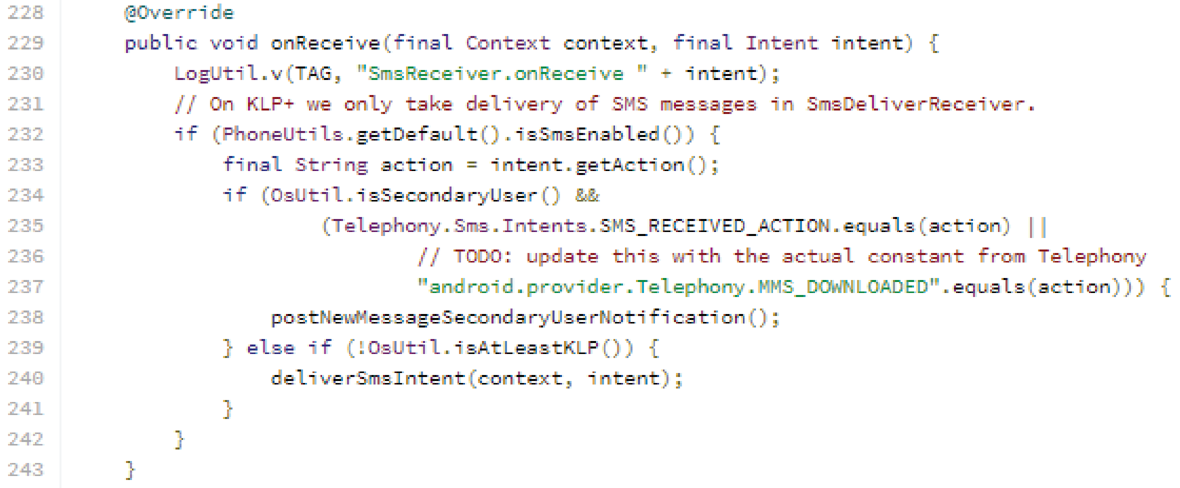
<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>



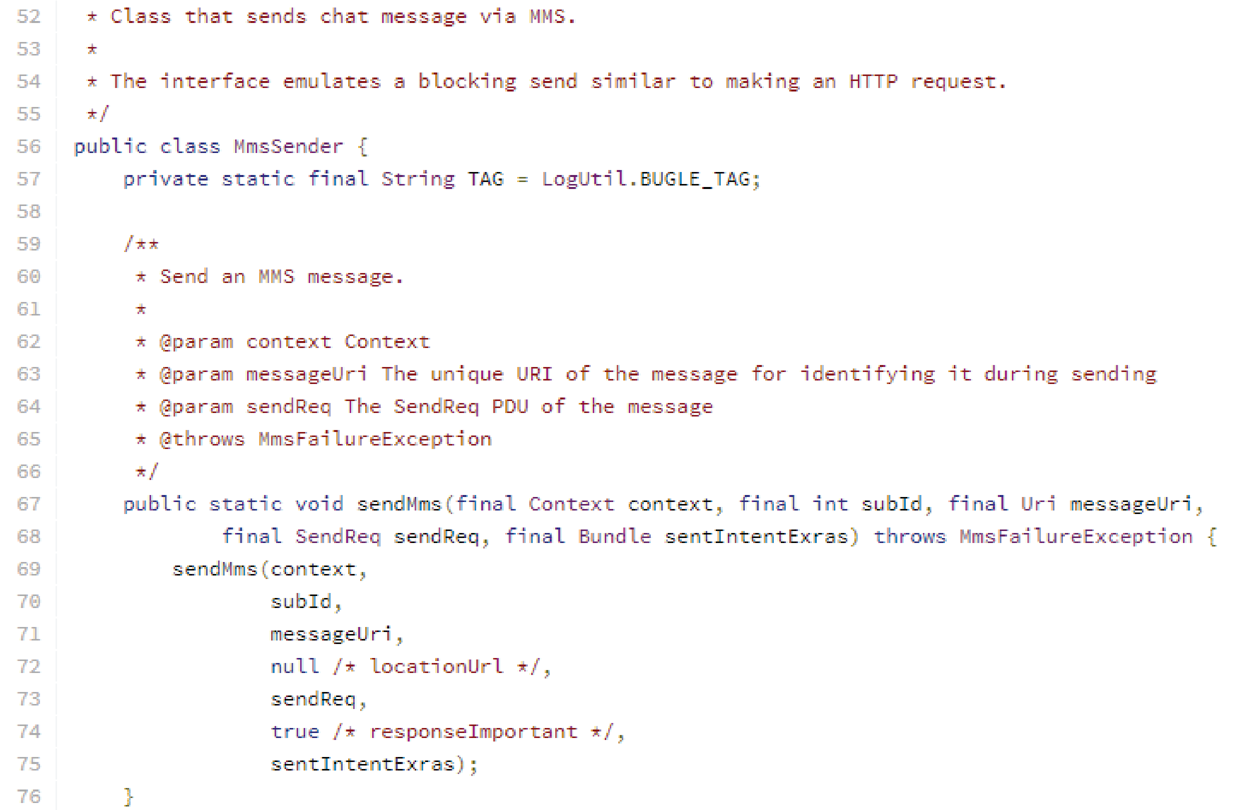
```
public static void deliverSmsMessages(final Context context, final int subId,
        final int errorCode, final android.telephony.SmsMessage[] messages) {
    final ContentValues messageValues =
            MmsUtils.parseReceivedSmsMessage(context, messages, errorCode);

    LogUtil.v(TAG, "SmsReceiver.deliverSmsMessages");

    final long nowInMillis =  System.currentTimeMillis();
    final long receivedTimestampMs = MmsUtils.getMessageDate(messages[0], nowInMillis);

    messageValues.put(Sms.Inbox.DATE, receivedTimestampMs);
    // Default to unread and unseen for us but ReceiveSmsMessageAction will override
    // seen for the telephony db.
    messageValues.put(Sms.Inbox.READ, 0);
    messageValues.put(Sms.Inbox.SEEN, 0);
    if (OsUtil.isAtLeastL_MR1()) {
        messageValues.put(Sms.SUBSCRIPTION_ID, subId);
    }

    if (messages[0].getMessageClass() == android.telephony.SmsMessage.MessageClass.CLASS_0 ||
            DebugUtils.debugClassZeroSmsEnabled()) {
        Factory.get().getUIIntents().launchClassZeroActivity(context, messageValues);
    } else {
        final ReceiveSmsMessageAction action = new ReceiveSmsMessageAction(messageValues);
        action.start();
    }
}
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/receiver/SmsReceiver.java
</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>
<pre>
@Override
public void onReceive(final Context context, final Intent intent) {
    LogUtil.v(TAG, "SmsReceiver.onReceive " + intent);
    // On KLP+ we only take delivery of SMS messages in SmsDeliverReceiver.
    if (PhoneUtils.getDefault().isSmsEnabled()) {
        final String action = intent.getAction();
        if (OsUtil.isSecondaryUser() &&
                (Telephony.Sms.Intents.SMS_RECEIVED_ACTION.equals(action) ||
                        // TODO: update this with the actual constant from Telephony
                        "android.provider.Telephony.MMS_DOWNLOADED".equals(action))) {
            postNewMessageSecondaryUserNotification();
        } else if (!OsUtil.isAtLeastKLP()) {
            deliverSmsIntent(context, intent);
        }
    }
}
</pre>
https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/receiver/SmsReceiver.java</td></tr>
</table>

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>



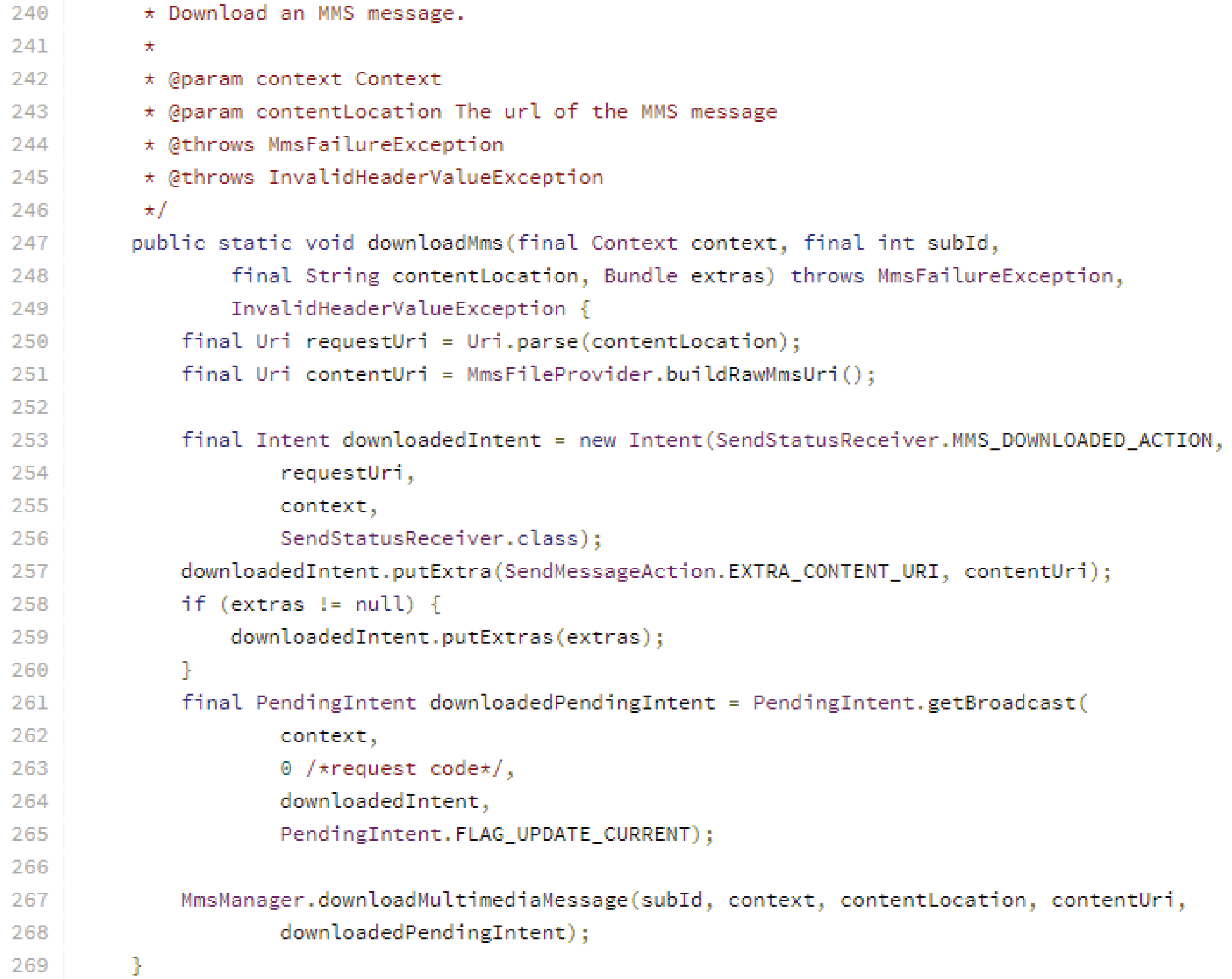
```
 * Class that sends chat message via MMS.
 *
 * The interface emulates a blocking send similar to making an HTTP request.
 */
public class MmsSender {
    private static final String TAG = LogUtil.BUGLE_TAG;

    /**
     * Send an MMS message.
     *
     * @param context Context
     * @param messageUri The unique URI of the message for identifying it during sending
     * @param sendReq The SendReq PDU of the message
     * @throws MmsFailureException
     */
    public static void sendMms(final Context context, final int subId, final Uri messageUri,
            final SendReq sendReq, final Bundle sentIntentExras) throws MmsFailureException {
        sendMms(context,
                subId,
                messageUri,
                null /* locationUrl */,
                sendReq,
                true /* responseImportant */,
                sentIntentExras);
    }
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/MmsSender.java

</td></tr>
</table>

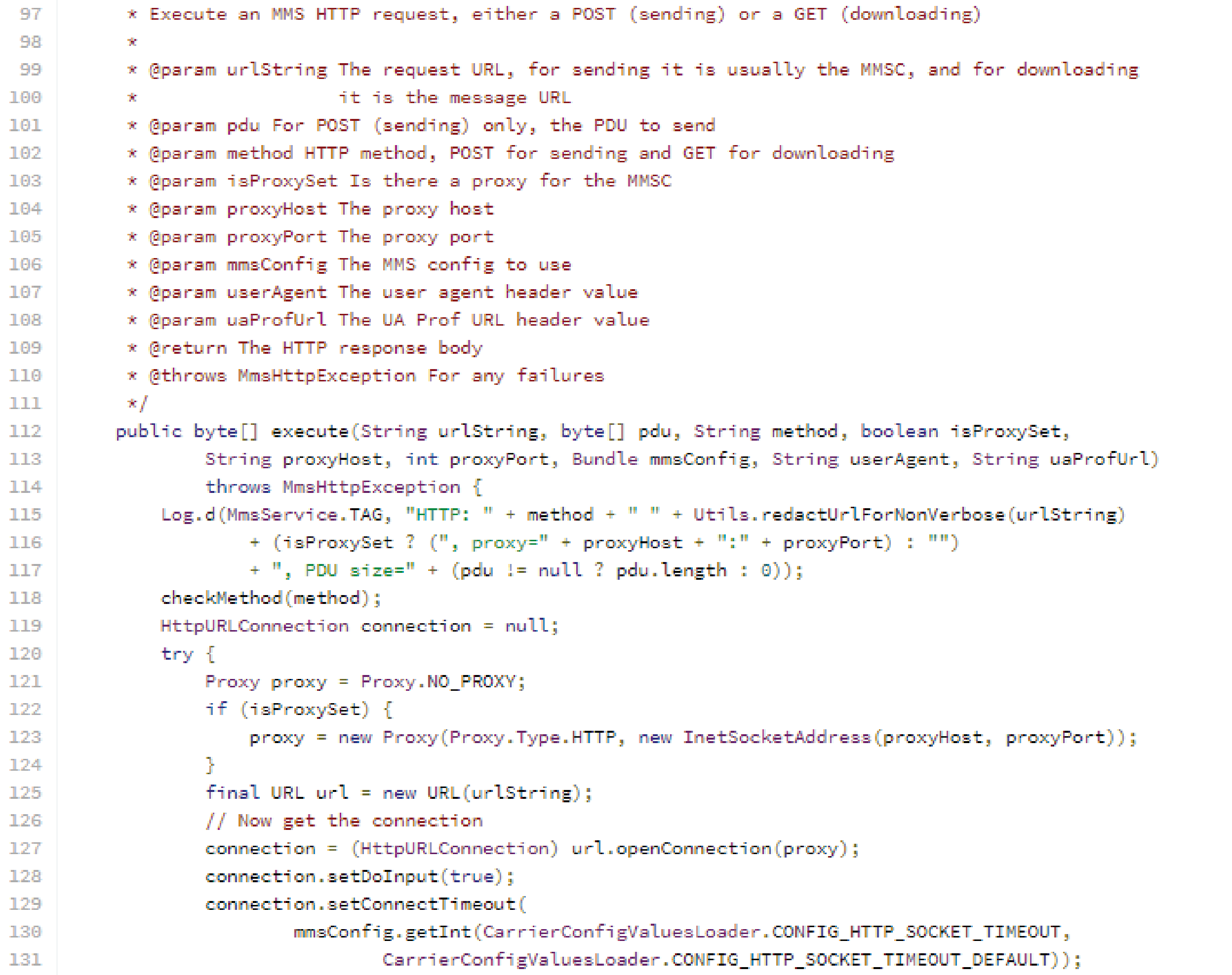
Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products



```
     * Download an MMS message.
     *
     * @param context Context
     * @param contentLocation The url of the MMS message
     * @throws MmsFailureException
     * @throws InvalidHeaderValueException
     */
    public static void downloadMms(final Context context, final int subId,
            final String contentLocation, Bundle extras) throws MmsFailureException,
            InvalidHeaderValueException {
        final Uri requestUri = Uri.parse(contentLocation);
        final Uri contentUri = MmsFileProvider.buildRawMmsUri();

        final Intent downloadedIntent = new Intent(SendStatusReceiver.MMS_DOWNLOADED_ACTION,
                requestUri,
                context,
                SendStatusReceiver.class);
        downloadedIntent.putExtra(SendMessageAction.EXTRA_CONTENT_URI, contentUri);
        if (extras != null) {
            downloadedIntent.putExtras(extras);
        }
        final PendingIntent downloadedPendingIntent = PendingIntent.getBroadcast(
                context,
                0 /*request code*/,
                downloadedIntent,
                PendingIntent.FLAG_UPDATE_CURRENT);

        MmsManager.downloadMultimediaMessage(subId, context, contentLocation, contentUri,
                downloadedPendingIntent);
    }
```

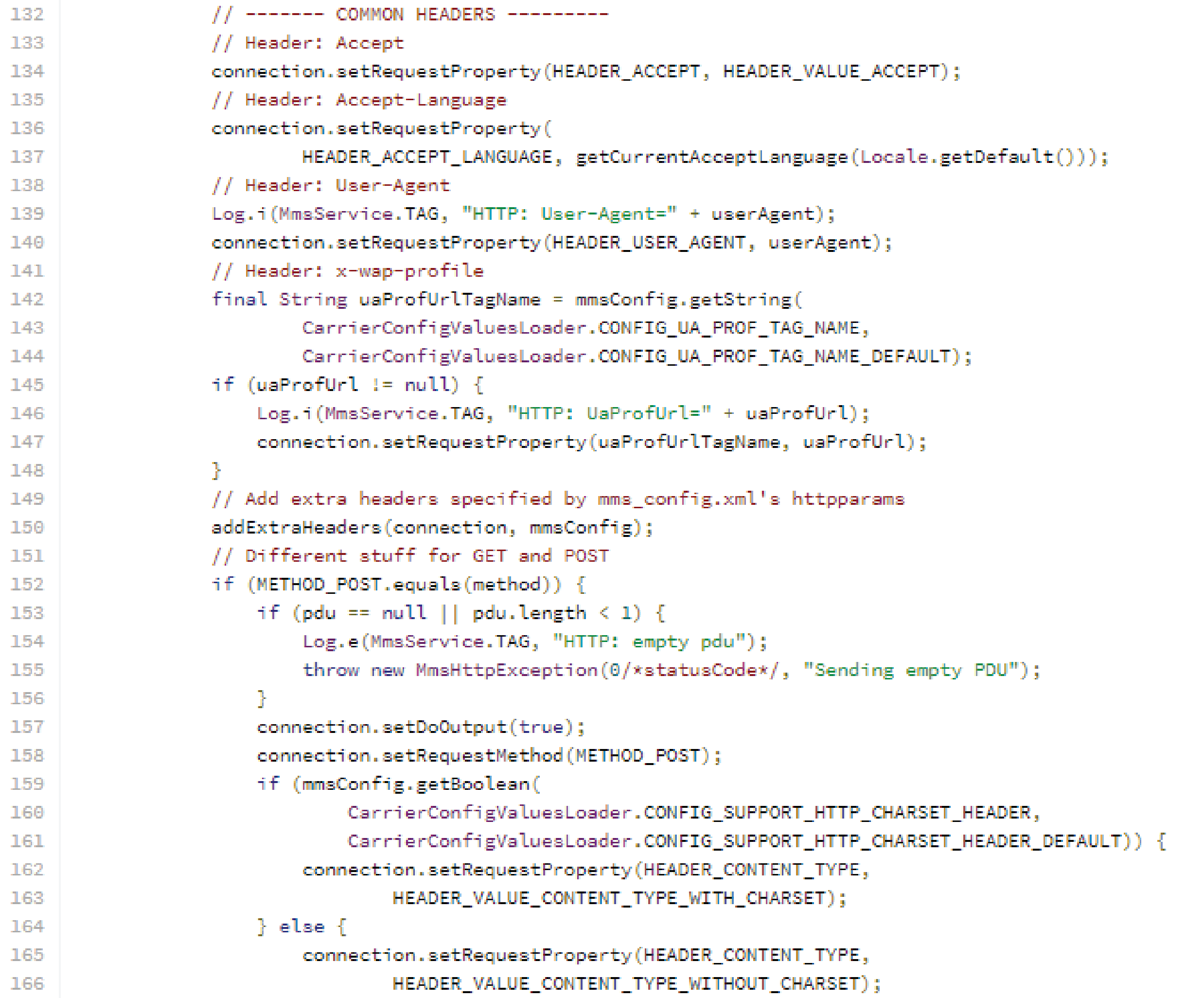
https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/com/android/messaging/sms/MmsSender.java

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
     * Execute an MMS HTTP request, either a POST (sending) or a GET (downloading)
     *
     * @param urlString The request URL, for sending it is usually the MMSC, and for downloading
     *                  it is the message URL
     * @param pdu For POST (sending) only, the PDU to send
     * @param method HTTP method, POST for sending and GET for downloading
     * @param isProxySet Is there a proxy for the MMSC
     * @param proxyHost The proxy host
     * @param proxyPort The proxy port
     * @param mmsConfig The MMS config to use
     * @param userAgent The user agent header value
     * @param uaProfUrl The UA Prof URL header value
     * @return The HTTP response body
     * @throws MmsHttpException For any failures
     */
    public byte[] execute(String urlString, byte[] pdu, String method, boolean isProxySet,
            String proxyHost, int proxyPort, Bundle mmsConfig, String userAgent, String uaProfUrl)
            throws MmsHttpException {
        Log.d(MmsService.TAG, "HTTP: " + method + " " + Utils.redactUrlForNonVerbose(urlString)
                + (isProxySet ? (", proxy=" + proxyHost + ":" + proxyPort) : "")
                + ", PDU size=" + (pdu != null ? pdu.length : 0));
        checkMethod(method);
        HttpURLConnection connection = null;
        try {
            Proxy proxy = Proxy.NO_PROXY;
            if (isProxySet) {
                proxy = new Proxy(Proxy.Type.HTTP, new InetSocketAddress(proxyHost, proxyPort));
            }
            final URL url = new URL(urlString);
            // Now get the connection
            connection = (HttpURLConnection) url.openConnection(proxy);
            connection.setDoInput(true);
            connection.setConnectTimeout(
                    mmsConfig.getInt(CarrierConfigValuesLoader.CONFIG_HTTP_SOCKET_TIMEOUT,
                            CarrierConfigValuesLoader.CONFIG_HTTP_SOCKET_TIMEOUT_DEFAULT));
```

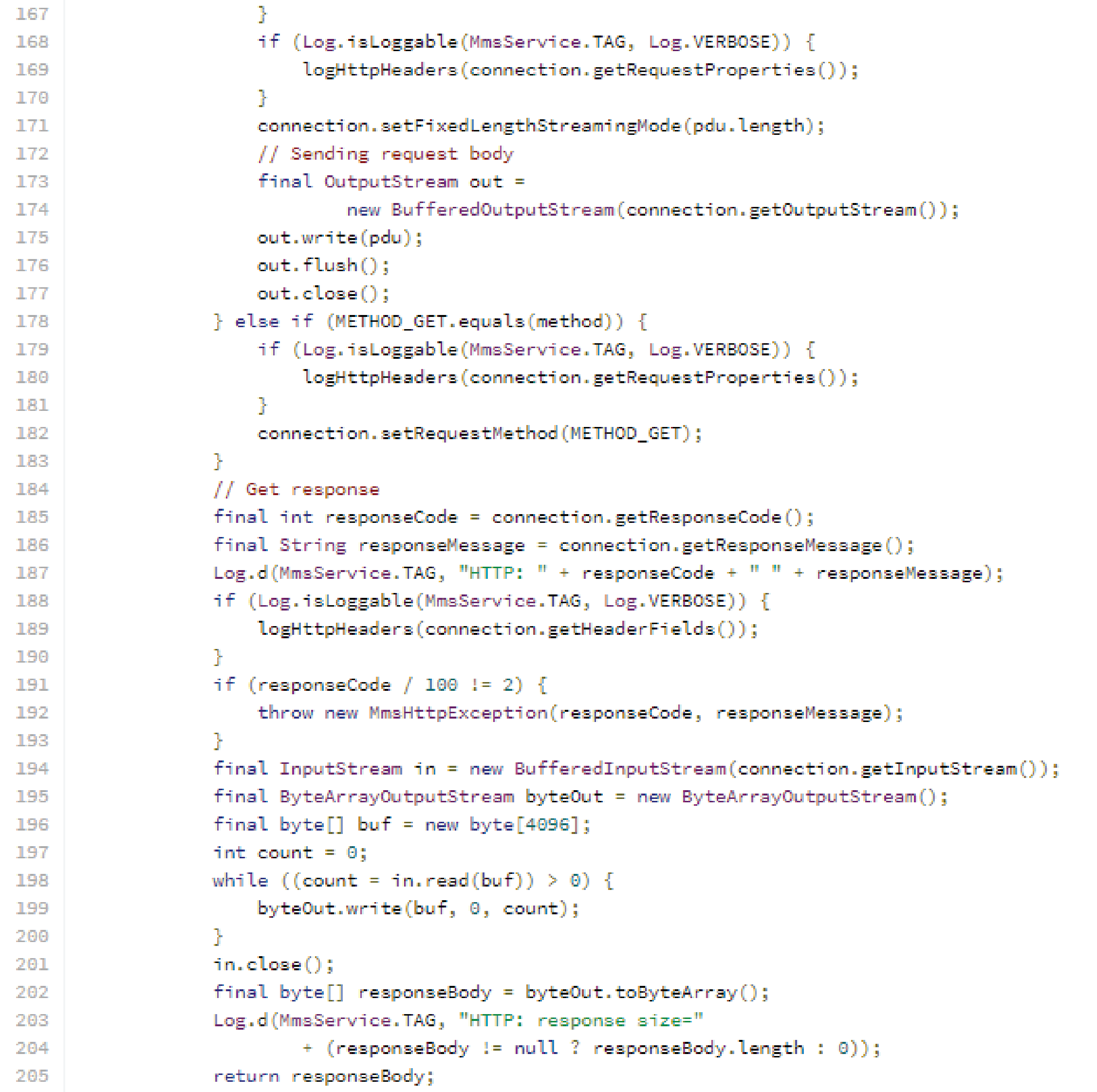
https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/android/support/v7/mms/MmsHttpClient.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>


```
// ------- COMMON HEADERS ---------
// Header: Accept
connection.setRequestProperty(HEADER_ACCEPT, HEADER_VALUE_ACCEPT);
// Header: Accept-Language
connection.setRequestProperty(
        HEADER_ACCEPT_LANGUAGE, getCurrentAcceptLanguage(Locale.getDefault()));
// Header: User-Agent
Log.i(MmsService.TAG, "HTTP: User-Agent=" + userAgent);
connection.setRequestProperty(HEADER_USER_AGENT, userAgent);
// Header: x-wap-profile
final String uaProfUrlTagName = mmsConfig.getString(
        CarrierConfigValuesLoader.CONFIG_UA_PROF_TAG_NAME,
        CarrierConfigValuesLoader.CONFIG_UA_PROF_TAG_NAME_DEFAULT);
if (uaProfUrl != null) {
    Log.i(MmsService.TAG, "HTTP: UaProfUrl=" + uaProfUrl);
    connection.setRequestProperty(uaProfUrlTagName, uaProfUrl);
}
// Add extra headers specified by mms_config.xml's httpparams
addExtraHeaders(connection, mmsConfig);
// Different stuff for GET and POST
if (METHOD_POST.equals(method)) {
    if (pdu == null || pdu.length < 1) {
        Log.e(MmsService.TAG, "HTTP: empty pdu");
        throw new MmsHttpException(0/*statusCode*/, "Sending empty PDU");
    }
    connection.setDoOutput(true);
    connection.setRequestMethod(METHOD_POST);
    if (mmsConfig.getBoolean(
            CarrierConfigValuesLoader.CONFIG_SUPPORT_HTTP_CHARSET_HEADER,
            CarrierConfigValuesLoader.CONFIG_SUPPORT_HTTP_CHARSET_HEADER_DEFAULT)) {
        connection.setRequestProperty(HEADER_CONTENT_TYPE,
                HEADER_VALUE_CONTENT_TYPE_WITH_CHARSET);
    } else {
        connection.setRequestProperty(HEADER_CONTENT_TYPE,
                HEADER_VALUE_CONTENT_TYPE_WITHOUT_CHARSET);
```

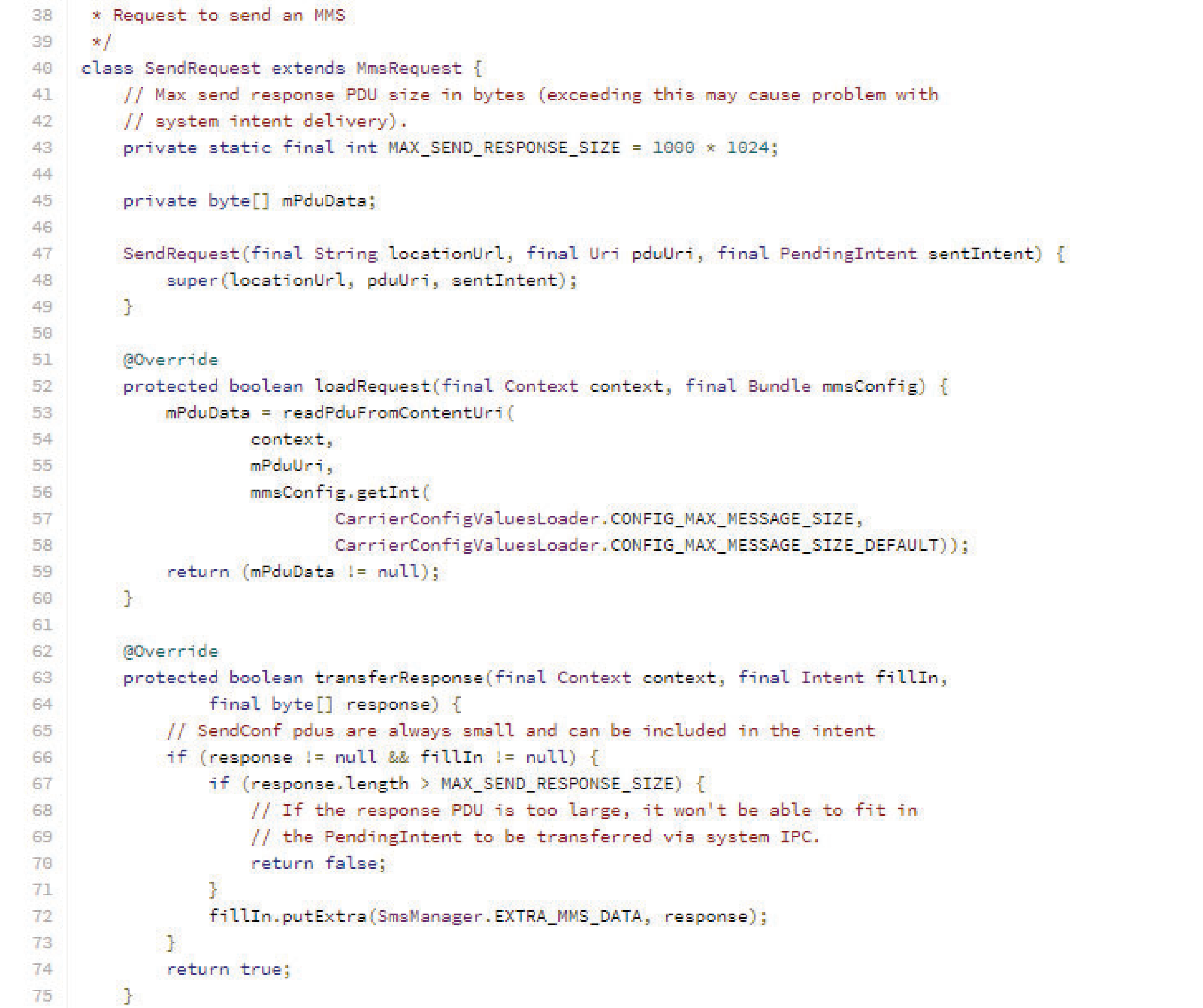
https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/android/support/v7/mms/MmsHttpClient.java
</td></tr>
</table>

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	

```
            }
            if (Log.isLoggable(MmsService.TAG, Log.VERBOSE)) {
                logHttpHeaders(connection.getRequestProperties());
            }
            connection.setFixedLengthStreamingMode(pdu.length);
            // Sending request body
            final OutputStream out =
                    new BufferedOutputStream(connection.getOutputStream());
            out.write(pdu);
            out.flush();
            out.close();
        } else if (METHOD_GET.equals(method)) {
            if (Log.isLoggable(MmsService.TAG, Log.VERBOSE)) {
                logHttpHeaders(connection.getRequestProperties());
            }
            connection.setRequestMethod(METHOD_GET);
        }
        // Get response
        final int responseCode = connection.getResponseCode();
        final String responseMessage = connection.getResponseMessage();
        Log.d(MmsService.TAG, "HTTP: " + responseCode + " " + responseMessage);
        if (Log.isLoggable(MmsService.TAG, Log.VERBOSE)) {
            logHttpHeaders(connection.getHeaderFields());
        }
        if (responseCode / 100 != 2) {
            throw new MmsHttpException(responseCode, responseMessage);
        }
        final InputStream in = new BufferedInputStream(connection.getInputStream());
        final ByteArrayOutputStream byteOut = new ByteArrayOutputStream();
        final byte[] buf = new byte[4096];
        int count = 0;
        while ((count = in.read(buf)) > 0) {
            byteOut.write(buf, 0, count);
        }
        in.close();
        final byte[] responseBody = byteOut.toByteArray();
        Log.d(MmsService.TAG, "HTTP: response size="
                + (responseBody != null ? responseBody.length : 0));
        return responseBody;
```

https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/android/support/v7/mms/MmsHttpClient.java

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>


```java
 * Request to send an MMS
 */
class SendRequest extends MmsRequest {
    // Max send response PDU size in bytes (exceeding this may cause problem with
    // system intent delivery).
    private static final int MAX_SEND_RESPONSE_SIZE = 1000 * 1024;

    private byte[] mPduData;

    SendRequest(final String locationUrl, final Uri pduUri, final PendingIntent sentIntent) {
        super(locationUrl, pduUri, sentIntent);
    }

    @Override
    protected boolean loadRequest(final Context context, final Bundle mmsConfig) {
        mPduData = readPduFromContentUri(
                context,
                mPduUri,
                mmsConfig.getInt(
                        CarrierConfigValuesLoader.CONFIG_MAX_MESSAGE_SIZE,
                        CarrierConfigValuesLoader.CONFIG_MAX_MESSAGE_SIZE_DEFAULT));
        return (mPduData != null);
    }

    @Override
    protected boolean transferResponse(final Context context, final Intent fillIn,
            final byte[] response) {
        // SendConf pdus are always small and can be included in the intent
        if (response != null && fillIn != null) {
            if (response.length > MAX_SEND_RESPONSE_SIZE) {
                // If the response PDU is too large, it won't be able to fit in
                // the PendingIntent to be transferred via system IPC.
                return false;
            }
            fillIn.putExtra(SmsManager.EXTRA_MMS_DATA, response);
        }
        return true;
    }
```
</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

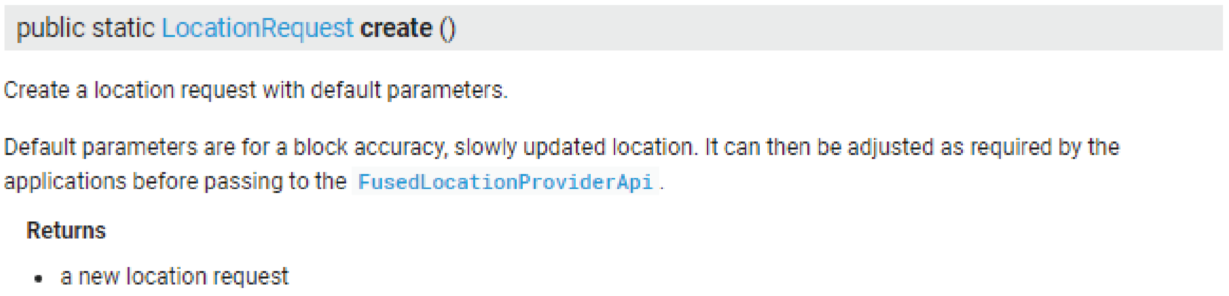
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	https://android.googlesource.com/platform/packages/apps/Messaging/+/nougat-mr1-release/src/android/support/v7/mms/SendRequest.java  public static LocationRequest **create** () Create a location request with default parameters. Default parameters are for a block accuracy, slowly updated location. It can then be adjusted as required by the applications before passing to the `FusedLocationProviderApi` . **Returns** - a new location request https://developers.google.com/android/reference/com/google/android/gms/location/LocationRequest

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

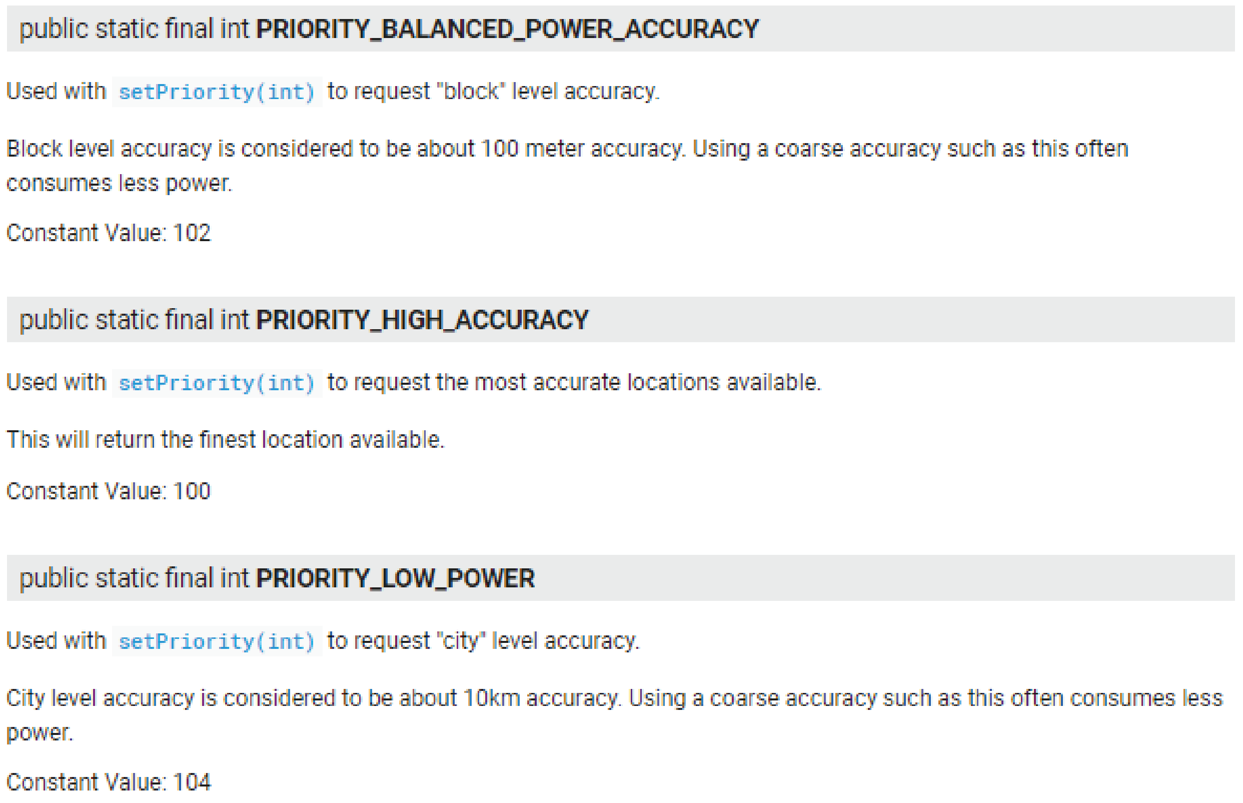
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public static final int **PRIORITY_BALANCED_POWER_ACCURACY** Used with `setPriority(int)` to request "block" level accuracy. Block level accuracy is considered to be about 100 meter accuracy. Using a coarse accuracy such as this often consumes less power. Constant Value: 102 public static final int **PRIORITY_HIGH_ACCURACY** Used with `setPriority(int)` to request the most accurate locations available. This will return the finest location available. Constant Value: 100 public static final int **PRIORITY_LOW_POWER** Used with `setPriority(int)` to request "city" level accuracy. City level accuracy is considered to be about 10km accuracy. Using a coarse accuracy such as this often consumes less power. Constant Value: 104 https://developers.google.com/android/reference/com/google/android/gms/location/LocationRequest

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

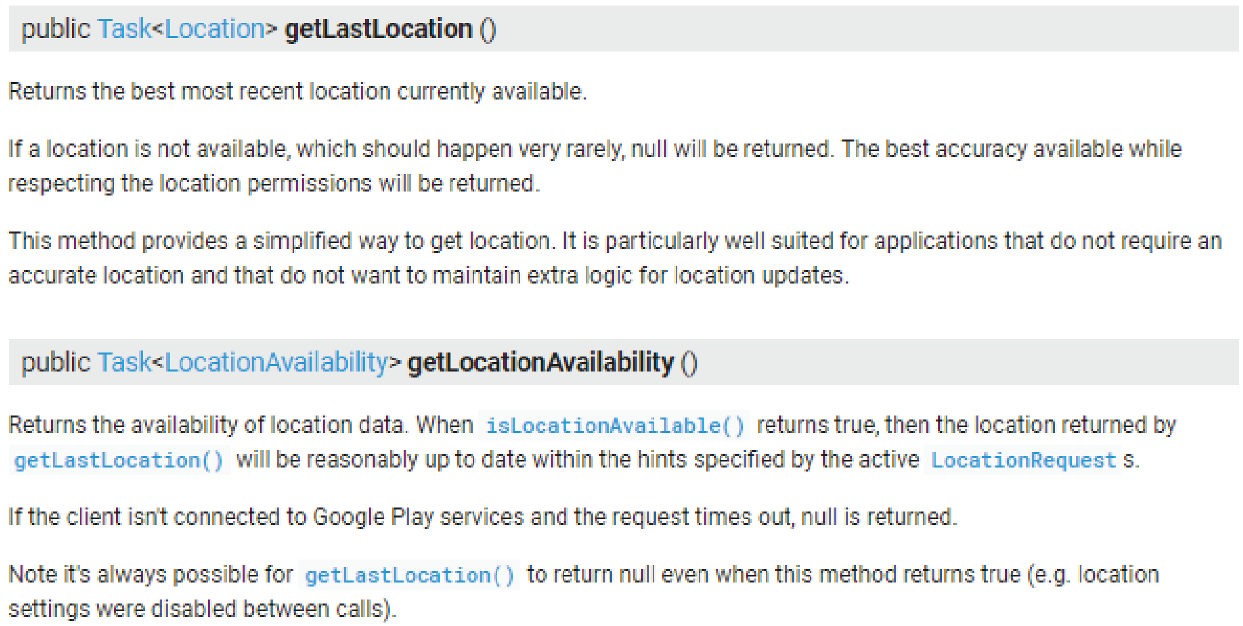
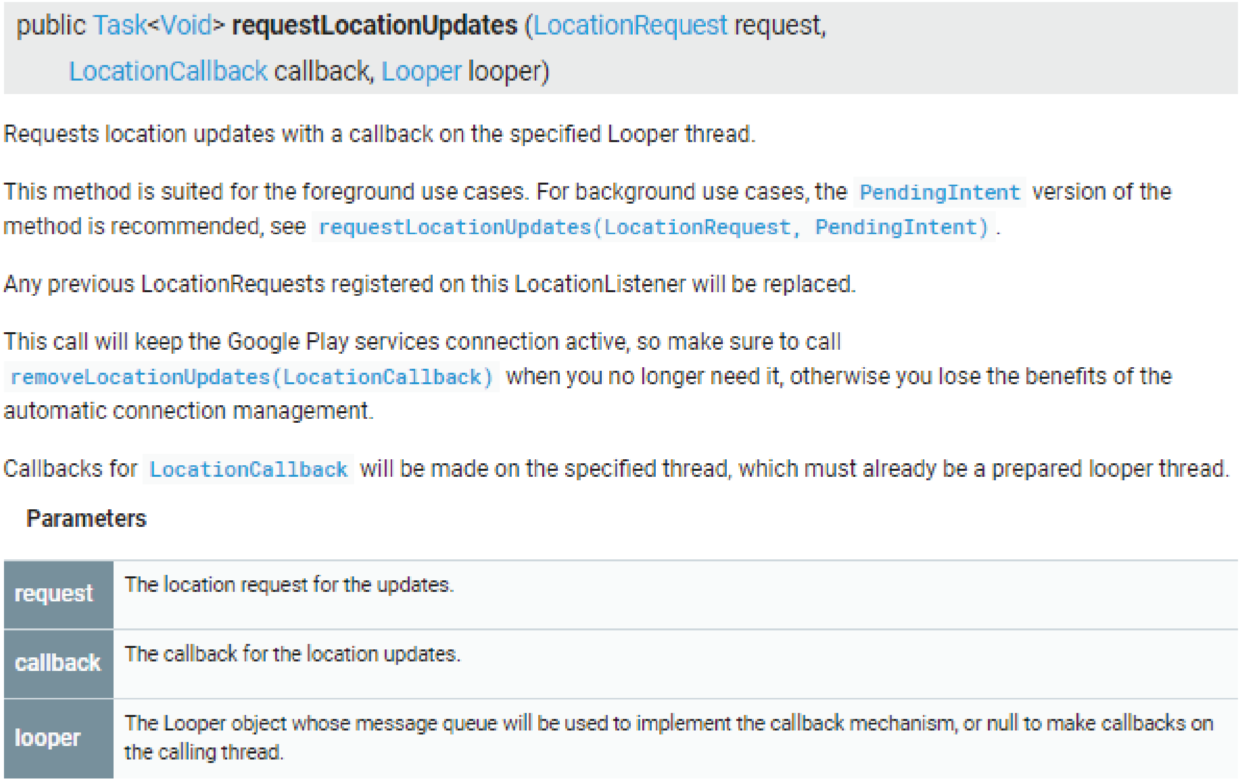
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public Task<Location> **getLastLocation** () Returns the best most recent location currently available. If a location is not available, which should happen very rarely, null will be returned. The best accuracy available while respecting the location permissions will be returned. This method provides a simplified way to get location. It is particularly well suited for applications that do not require an accurate location and that do not want to maintain extra logic for location updates. public Task<LocationAvailability> **getLocationAvailability** () Returns the availability of location data. When `isLocationAvailable()` returns true, then the location returned by `getLastLocation()` will be reasonably up to date within the hints specified by the active `LocationRequest` s. If the client isn't connected to Google Play services and the request times out, null is returned. Note it's always possible for `getLastLocation()` to return null even when this method returns true (e.g. location settings were disabled between calls). https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>


public Task<Void> **requestLocationUpdates** (LocationRequest request, LocationCallback callback, Looper looper)

Requests location updates with a callback on the specified Looper thread.

This method is suited for the foreground use cases. For background use cases, the `PendingIntent` version of the method is recommended, see `requestLocationUpdates(LocationRequest, PendingIntent)`.

Any previous LocationRequests registered on this LocationListener will be replaced.

This call will keep the Google Play services connection active, so make sure to call `removeLocationUpdates(LocationCallback)` when you no longer need it, otherwise you lose the benefits of the automatic connection management.

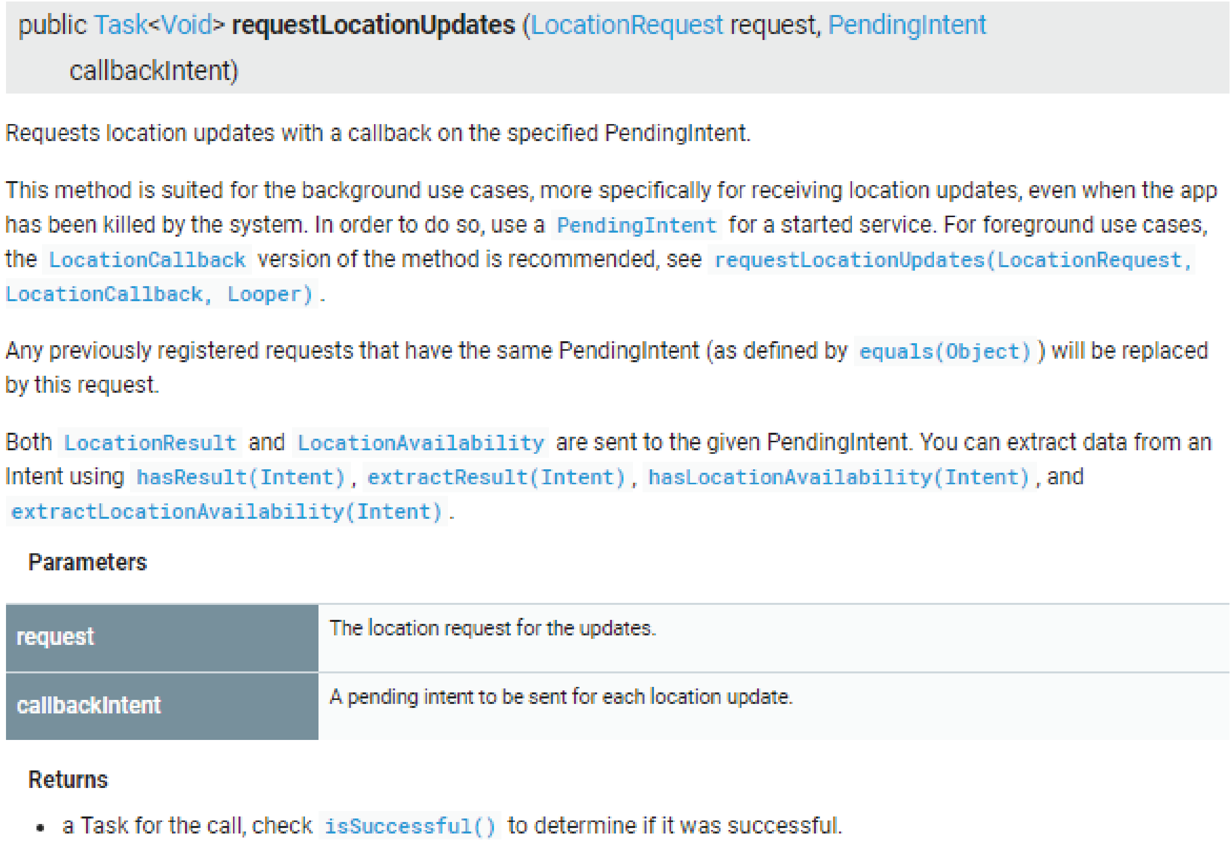
Callbacks for `LocationCallback` will be made on the specified thread, which must already be a prepared looper thread.

Parameters

request	The location request for the updates.
callback	The callback for the location updates.
looper	The Looper object whose message queue will be used to implement the callback mechanism, or null to make callbacks on the calling thread.

https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient
</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>


public Task<Void> **requestLocationUpdates** (LocationRequest request, PendingIntent callbackIntent)

Requests location updates with a callback on the specified PendingIntent.

This method is suited for the background use cases, more specifically for receiving location updates, even when the app has been killed by the system. In order to do so, use a `PendingIntent` for a started service. For foreground use cases, the `LocationCallback` version of the method is recommended, see `requestLocationUpdates(LocationRequest, LocationCallback, Looper)`.

Any previously registered requests that have the same PendingIntent (as defined by `equals(Object)`) will be replaced by this request.

Both `LocationResult` and `LocationAvailability` are sent to the given PendingIntent. You can extract data from an Intent using `hasResult(Intent)`, `extractResult(Intent)`, `hasLocationAvailability(Intent)`, and `extractLocationAvailability(Intent)`.

Parameters

<table>
<tr><td><b>request</b></td><td>The location request for the updates.</td></tr>
<tr><td><b>callbackIntent</b></td><td>A pending intent to be sent for each location update.</td></tr>
</table>

Returns

- a Task for the call, check `isSuccessful()` to determine if it was successful.

https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient
</td></tr>
</table>

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

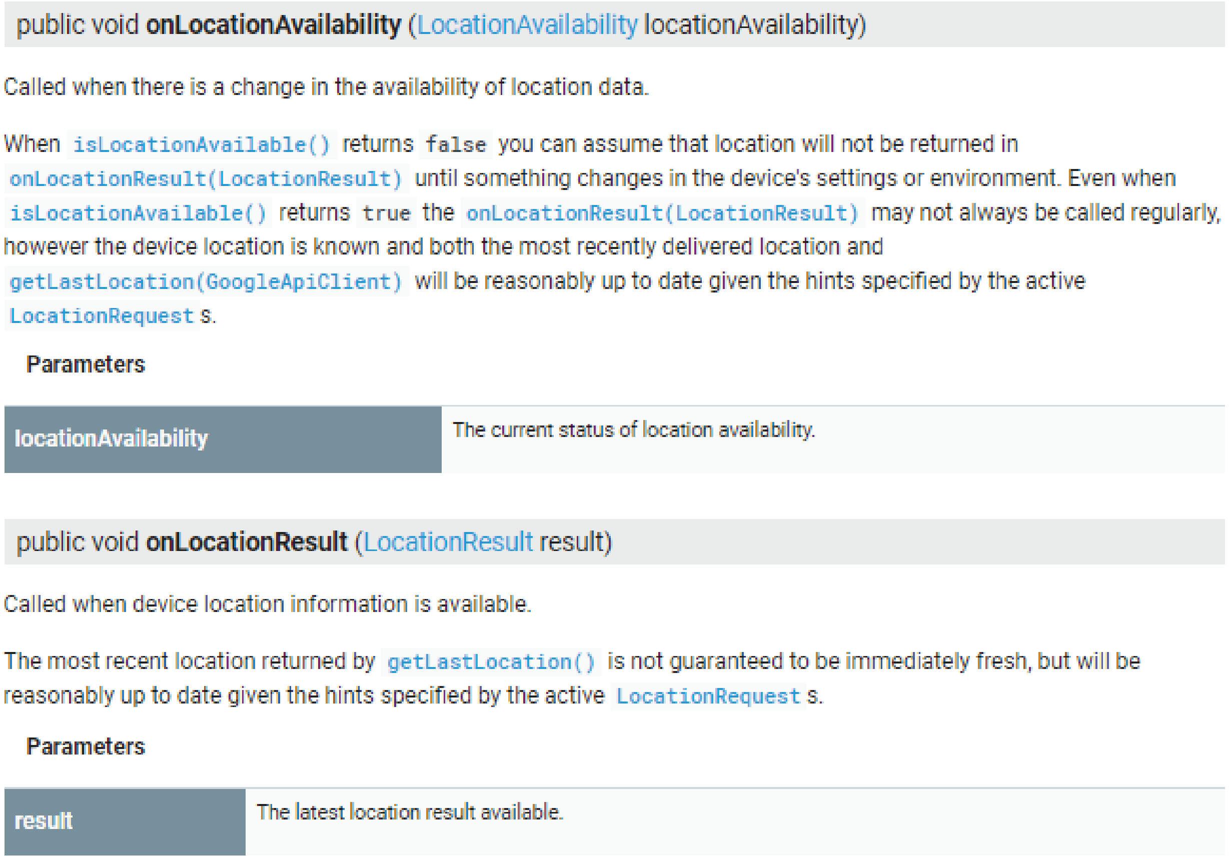
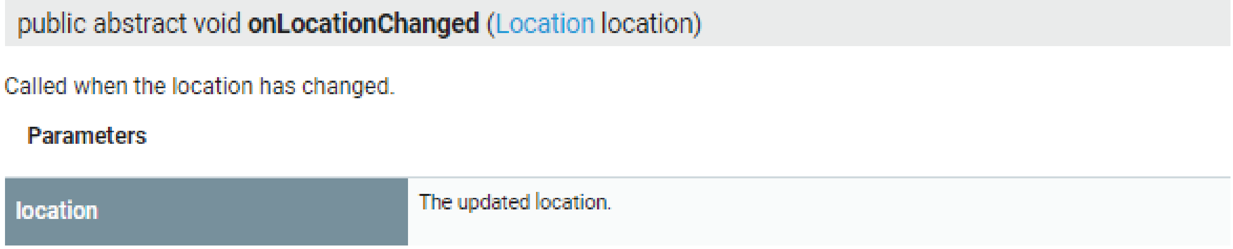
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public void **onLocationAvailability** (LocationAvailability locationAvailability) Called when there is a change in the availability of location data. When `isLocationAvailable()` returns `false` you can assume that location will not be returned in `onLocationResult(LocationResult)` until something changes in the device's settings or environment. Even when `isLocationAvailable()` returns `true` the `onLocationResult(LocationResult)` may not always be called regularly, however the device location is known and both the most recently delivered location and `getLastLocation(GoogleApiClient)` will be reasonably up to date given the hints specified by the active `LocationRequest` s. **Parameters** **locationAvailability** The current status of location availability. public void **onLocationResult** (LocationResult result) Called when device location information is available. The most recent location returned by `getLastLocation()` is not guaranteed to be immediately fresh, but will be reasonably up to date given the hints specified by the active `LocationRequest` s. **Parameters** **result** The latest location result available. https://developers.google.com/android/reference/com/google/android/gms/location/LocationCallback  public abstract void **onLocationChanged** (Location location) Called when the location has changed. **Parameters** **location** The updated location.

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

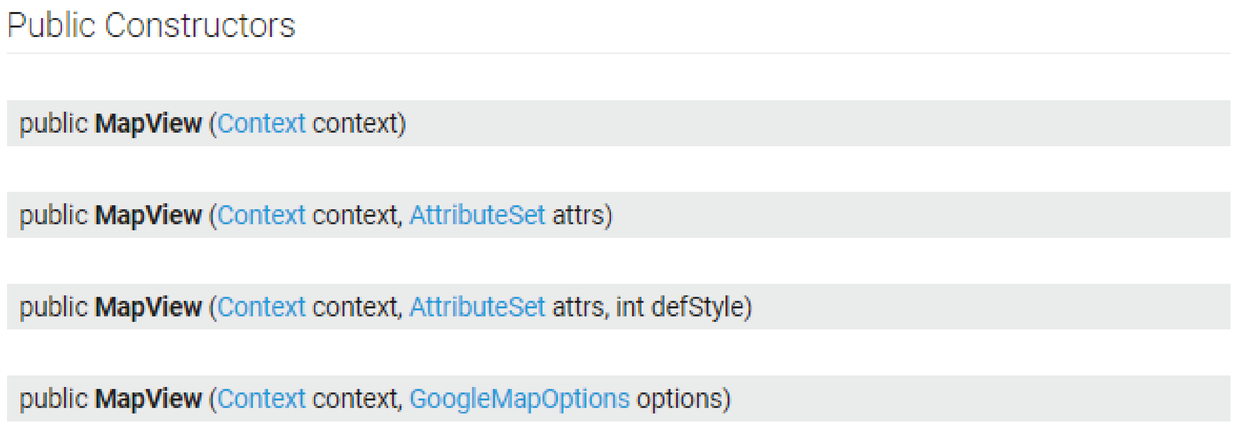
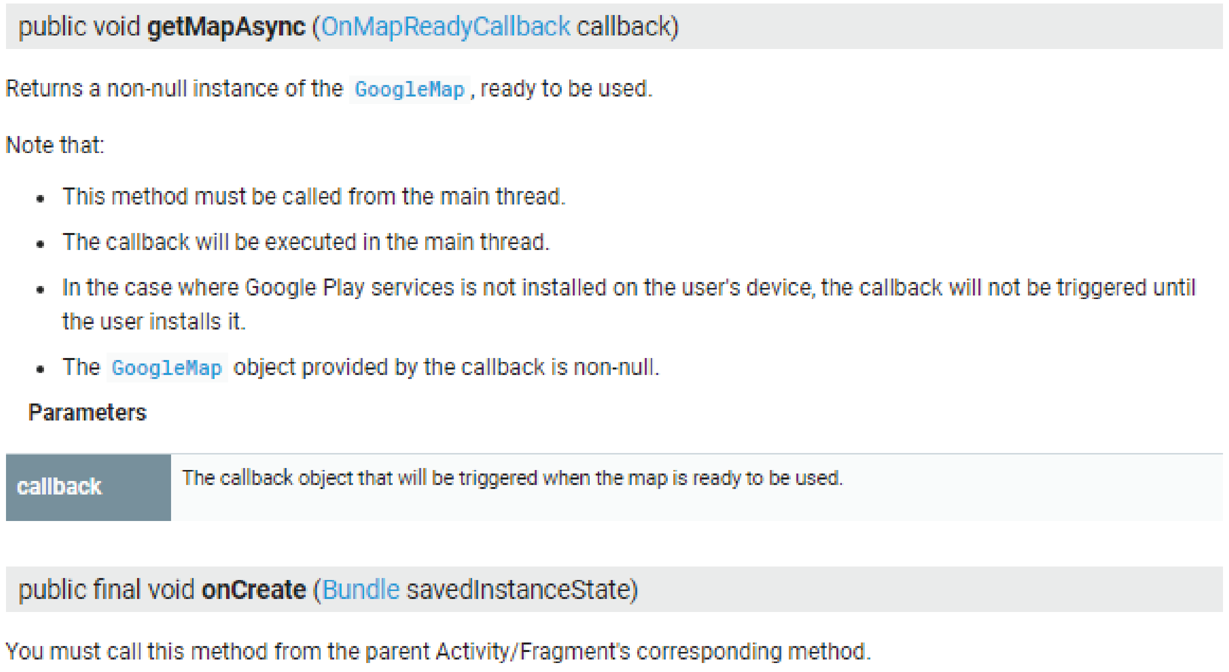
US9467838	Exemplary Supporting Evidence Regarding Accused Products
	https://developers.google.com/android/reference/com/google/android/gms/location/LocationListener  Public Constructors public **MapView** (Context context) public **MapView** (Context context, AttributeSet attrs) public **MapView** (Context context, AttributeSet attrs, int defStyle) public **MapView** (Context context, GoogleMapOptions options) https://developers.google.com/android/reference/com/google/android/gms/maps/MapView

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
	 public void **getMapAsync** (OnMapReadyCallback callback) Returns a non-null instance of the `GoogleMap`, ready to be used. Note that: • This method must be called from the main thread. • The callback will be executed in the main thread. • In the case where Google Play services is not installed on the user's device, the callback will not be triggered until the user installs it. • The `GoogleMap` object provided by the callback is non-null. **Parameters** callback: The callback object that will be triggered when the map is ready to be used. public final void **onCreate** (Bundle savedInstanceState) You must call this method from the parent Activity/Fragment's corresponding method. https://developers.google.com/android/reference/com/google/android/gms/maps/MapView
2. The method of claim 1, wherein the message is transmitted to the first server.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, wherein the message is transmitted to the first server. See, e.g. 1B and 1C above.
3. The method of claim 1, wherein the group comprises a plurality of group members permitted to communicate with each	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, wherein the group comprises a plurality of group members permitted to communicate with each other via the communication network. See, e.g. 1B above.

US9467838	Exemplary Supporting Evidence Regarding Accused Products
other via the communication network.	
4. The method of claim 1, wherein sending the third data to the selected one or more second devices via the first server comprises using an Internet Protocol to send the third data to the first server.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, wherein sending the third data to the selected one or more second devices via the first server comprises using an Internet Protocol to send the third data to the first server. See, e.g. 1H above. For example, contacting via chat, messaging, email, voice, and/or over applications such as Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome each include sending data over IP. For example, playing a sound and sending a message includes sending data over IP.
5. The method of claim 4, wherein the first device does not have access to respective Internet Protocol addresses of the one or more second devices included in the group.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 4, wherein the first device does not have access to respective Internet Protocol addresses of the one or more second devices included in the group. See, e.g. 1H above. Upon information and belief, sending data from a first device contacting via chat, messaging, email, voice, and/or over applications such as Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome does not require the IP address of the recipient device.
6. The method of claim 1, wherein the third data include a short message service message, a text message, an image, or a video.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, wherein the third data include a short message service message, a text message, an image, or a video. See, e.g. 1H above. For example, playing a sound and sending a message includes sending the third. For example, contacting via chat, messaging, email, voice, and/or over applications such as Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome each include sending data over IP. Upon information and belief, contacting via message includes SMS and text messages. Upon information and belief, contacting via Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome includes messaging text, images, and video.
7. The method of claim	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to

US9467838	Exemplary Supporting Evidence Regarding Accused Products
6, wherein the video comprises a video clip.	the performance of the method of claim 6, wherein the video comprises a video clip. See, e.g. 1H above. For example, contact over Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome includes sending a video clip.
8. The method of claim 1, wherein the third data include a voice recording.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, wherein the third data include a voice recording. See, e.g. 1H above. For example, contacting via Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome, include sending a voice recording or audio recording. For example, playing a sound a second device including sending and playing a voice file. For example, Facetime includes sending a recording of a voice. For example, each Accused Product can send a recording. For example, each Accused Product can send a recording. https://www.techrepublic.com/article/how-to-send-audio-clips-via-sms-in-android/
9. The method of claim 1, wherein sending the third data to the selected one or more second devices comprises transmitting a text message to at least one of the selected one or more second devices using an Internet Protocol (IP).	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, wherein sending the third data to the selected one or more second devices comprises transmitting a text message to at least one of the selected one or more second devices using an Internet Protocol (IP). See, e.g. 1H above. For example, contacting Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome includes sending a text message over IP. For example, entering and sending a message to a device includes sending the text message over IP.
10. The method of claim 1, further comprising performing by the first device: identifying user interaction with the	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, further comprising performing by the first device: identifying user interaction with the interactive display selecting at least one of the second set of user-selectable symbols corresponding to at least one of the second devices and user interaction with the display specifying an action and, based thereon, initiating a phone call or phone conference with the at least one

US9467838	Exemplary Supporting Evidence Regarding Accused Products
interactive display selecting at least one of the second set of user-selectable symbols corresponding to at least one of the second devices and user interaction with the display specifying an action and, based thereon, initiating a phone call or phone conference with the at least one second device.	second device. See, e.g. 1H above. For example, Find My Device and Android Device Manager allow a first device to initiate a phone call or phone conference by sending a phone number to the second device. For example, Google Maps and its predecessors allow a user to call another user by presenting the user of the first device with the option to select the friend's phone number to initiate call and the option to initiate communication. For example, Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome each provide for initiating a call and/or conference.
11. The method of claim 1, further comprising performing by the first device: identifying user interaction with the interactive display selecting a particular user-selectable symbol corresponding to a particular second device and user interaction with the display specifying an action and, based thereon, initiating voice-over-IP (VOIP) communication with the particular second device.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, further comprising performing by the first device: identifying user interaction with the interactive display selecting a particular user-selectable symbol corresponding to a particular second device and user interaction with the display specifying an action and, based thereon, initiating voice-over-IP (VOIP) communication with the particular second device. See, e.g. 1H above. For example, Find My Device allows a first device to initiate a phone call or phone conference by sending a phone number to the second device. For example, Google Maps allows a user to call the friend by presenting the user of the first device with the option to select the friend's phone number to initiate call and the option to initiate communication. For example, Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome each provide for initiating VOIP communication with another device.
12. The method of claim	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to

US9467838	Exemplary Supporting Evidence Regarding Accused Products
1, further comprising performing by the first device: identifying user interaction with the interactive display selecting a particular user-selectable symbol corresponding to a particular second device and user interaction with the display specifying an action and, based thereon, initiating a data call with the particular second device.	the performance of the method of claim 1, further comprising performing by the first device: identifying user interaction with the interactive display selecting a particular user-selectable symbol corresponding to a particular second device and user interaction with the display specifying an action and, based thereon, initiating a data call with the particular second device. See, e.g. 1H above. For example, Find My Device allows a first device to initiate a phone call or phone conference by sending a phone number to the second device. For example, Google Maps allows a user to call the friend by presenting the user of the first device with the option to select the friend's phone number to initiate call and the option to initiate a Facetime session. For example, Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome each provide for initiating communication upon selection of a contact.
13. The method of claim 1, wherein the first device is a personal digital assistant (PDA) or a personal computer (PC).	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, wherein the first device is a personal digital assistant (PDA) or a personal computer (PC). See, e.g., 1P and 1A above. Upon information and belief, the Accused Products are forms of PDAs in that the functionality of a PDA has been subsumed into smartphones, tablets, and portable media players having the functionalities of a PDA. Upon information and belief, the Accused products are forms of personal computers in that the Accused Products are mobile or portable forms of personal computers. To the extent that it is necessary, AGIS submits that the Accused Products meet the claim limitation "the first device is a personal digital assistant (PDA) or a personal computer (PC)" under the doctrine of equivalents.
14. The method of claim 1, wherein the first device is a smart phone.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, wherein the first device is a smart phone. See, e.g., 1P and 1A above. For example, the Accused Products include a smartphone. To the extent that it is necessary, AGIS submits that tablets within the Accused Products meet the claim limitation "the first device is smartphone" under the doctrine of equivalents.
15. The method of claim 1, wherein the display of the first device is a touch	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, wherein the display of the first device is a touch screen display, and wherein the user interaction with the display selecting the one or more user-selectable symbols in the

US9467838	Exemplary Supporting Evidence Regarding Accused Products
screen display, and wherein the user interaction with the display selecting the one or more user-selectable symbols in the second set of symbols comprises touching the one or more user-selectable symbols in the second set of symbols.	second set of symbols comprises touching the one or more user-selectable symbols in the second set of symbols. See, e.g., 1P, 1A, and 1H above. For example, the Accused Devices are controlled by touch displays and selection of the symbol occurs on the touch display.
16. The method of claim 1, further comprising performing by the first device: sending updated location information comprising an updated location of the first device, the updated location information being sent based on passage of a predetermined time interval since sending previous location information comprising a previous location of the first device, displacement of the first device by a predetermined distance relative to a previous	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, further comprising performing by the first device: sending updated location information comprising an updated location of the first device, the updated location information being sent based on passage of a predetermined time interval since sending previous location information comprising a previous location of the first device, displacement of the first device by a predetermined distance relative to a previous location of the first device, or both. See, e.g., 1G above. For example, when a first device moves from a first to a second position, the symbol's position on the map is updated to reflect the change. Upon information and belief, the transmission of updated location information is based on a predetermined time passage or a predetermined distance change. For example the Accused Products utilize a location manager service such as LocationServices that monitors, among other things, periods of time that are predetermined, or displacement from a predetermined distance. This method is thus performed at least when a user passes a geographic threshold that results in a location update (e.g., "leaves" notification). For example, Android utilizes the following framework that meets this limitation as implemented on the Accused Products: https://developers.google.com/android/reference/com/google/android/gms/location/LocationServices https://developer.android.com/reference/android/location/LocationManager.html

US9467838	Exemplary Supporting Evidence Regarding Accused Products
location of the first device, or both.	e.g. requestLocationUpdates(String provider, long minTime, float minDistance, LocationListener listener) Register for location updates using the named provider, and a pending intent. Android also makes use of "geotagging" that invokes time-based updating.
17. The method of claim 1, further comprising performing by the first device: using a Global Positioning Satellite (GPS) receiver of the first device to obtain data indicative of the location of the first device, wherein sending the first location information to the first server comprises using an Internet Protocol (IP) to send the first location information to the first server.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, further comprising performing by the first device: using a Global Positioning Satellite (GPS) receiver of the first device to obtain data indicative of the location of the first device, wherein sending the first location information to the first server comprises using an Internet Protocol (IP) to send the first location information to the first server. See, e.g., 1P and 1A above regarding the hardware and software including GPS and A-GPS. See, e.g., 1C and 1D regarding the employment of location services and location sharing to obtain and send locations. For example, Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome use location services and location sharing to obtain locations and send those locations and transmit locations to a server.
18. The method of claim 17, wherein sending the first location information to the first server further comprises sending the first location information via the Internet.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 17, wherein sending the first location information to the first server further comprises sending the first location information via the Internet. See, e.g., 1P and 1A above regarding the hardware and software for transmitting data over the Internet. See, e.g., 1C and 1D regarding the employment of location services and location sharing to obtain and send locations over the Internet. For example, Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome use location services and location sharing to obtain locations and send those locations over the Internet for transmission to recipient devices/users.
19. The method of claim 1, wherein participating	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, wherein participating in the group further includes sending first

US9467838	Exemplary Supporting Evidence Regarding Accused Products
in the group further includes sending first status information to the first server and receiving second status information from the first server, wherein the first status information comprises data indicative of a battery level of the first device, a signal strength of a wireless signal of the first device, a status of a Global Positioning Satellite (GPS) receiver of the first device, or a combination thereof, and wherein the second status information comprises data indicative of one or more battery levels of the respective one or more second devices included in the group, one or more signal strengths of wireless signals of the respective one or more second devices included in the group, one or more	status information to the first server and receiving second status information from the first server, wherein the first status information comprises data indicative of a battery level of the first device, a signal strength of a wireless signal of the first device, a status of a Global Positioning Satellite (GPS) receiver of the first device, or a combination thereof, and wherein the second status information comprises data indicative of one or more battery levels of the respective one or more second devices included in the group, one or more signal strengths of wireless signals of the respective one or more second devices included in the group, one or more statuses of GPS receivers of the respective one or more second devices included in the group, or a combination thereof. See, e.g., 1A, 1C, and 1D. For example, Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome show status indicators for each member of the group, including an indicator of connectivity status of each member (see below). Upon information and belief, status metadata regarding the GPS receiver, wireless signals, device battery, and other status metadata are sent and received between the Accused Products. For example, the Accused Products each the ability to communicate battery level and wireless networks to servers, where that battery level and wireless networks are communicated to the first device. Additionally, the status of the GPS receivers is also communicated within the apps. For example, the second devices communicate the accuracy of their GPS signal, as well as status so that the first device can view both the accuracy of the signal (represented by a circle around the user) as well as the status (online vs. offline). Additionally, status of a GPS receiver may be communicated through notifications that communicate when a device has passed a location threshold.

US9467838	Exemplary Supporting Evidence Regarding Accused Products
statuses of GPS receivers of the respective one or more second devices included in the group, or a combination thereof.	
20. The method of claim 1, wherein the second georeferenced map data comprise a satellite image or aerial photograph.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, wherein the second georeferenced map data comprise a satellite image or aerial photograph. See, e.g., 1D and 1G. For example and shown below, Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome show maps in satellite and/or aerial formats.
21. The method of claim 1, wherein the spatial coordinates comprise latitude and longitude coordinates.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, wherein the spatial coordinates comprise latitude and longitude coordinates. See, e.g., 1C, 1D, and 1G. For example, the Accused Products present maps with, at least, latitude and longitude coordinates.
22[A]. The method of claim 1, further comprising identifying, by the first device, user interaction with the display selecting a particular user-selectable symbol positioned on the second georeferenced map and corresponding to a particular second device, wherein identifying the	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, further comprising identifying, by the first device, user interaction with the display selecting a particular user-selectable symbol positioned on the second georeferenced map and corresponding to a particular second device, wherein identifying the user interaction selecting the particular user-selectable symbol comprises: detecting user selection of a portion of the interactive display corresponding to a position on the second georeferenced map. See, e.g., 1G and 1H. For example, the user of the first device selects a symbol on the map by touching the display. The selection occurs on a touch display of the first device which includes logic for detecting the touch position and corresponding the touch positon to a position on the georeferenced map. See, e.g., 1P an 1A.

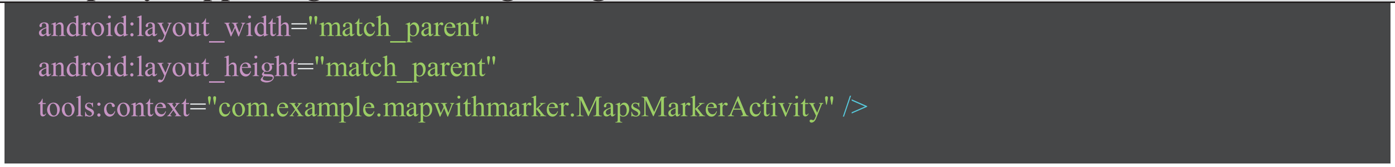
Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products
user interaction selecting the particular user-selectable symbol comprises: detecting user selection of a portion of the interactive display corresponding to a position on the second georeferenced map;	

[22B] based at least in part on coordinates of the selected position on the second georeferenced map and on the second georeferenced map data relating positions on the second georeferenced map to spatial coordinates, determining spatial coordinates of a location represented by the selected position on the second georeferenced map;	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of, based at least in part on coordinates of the selected position on the second georeferenced map and on the second georeferenced map data relating positions on the second georeferenced map to spatial coordinates, determining spatial coordinates of a location represented by the selected position on the second georeferenced map. See, e.g., 1P, 1A, 1G and 1H. Upon information and belief, the first device determines spatial coordinates representing selected positions on the map.
	For example, when markers, such as symbols, are added to maps, they are added based on longitude and latitude.
	Add a map
	Display a map, using the Google Maps Android API.
	Add a <fragment> element to your activity's layout file, activity_maps.xml. This element defines aSupportMapFragment to act as a container for the map and to provide access to the GoogleMap object. The tutorial uses the Android support library version of the map fragment, to ensure backward compatibility with earlier versions of the Android framework.
	`<fragment xmlns:android="http://schemas.android.com/apk/res/android"`
	`    xmlns:tools="http://schemas.android.com/tools"`
	`    android:id="@+id/map"`
	`    android:name="com.google.android.gms.maps.SupportMapFragment"`

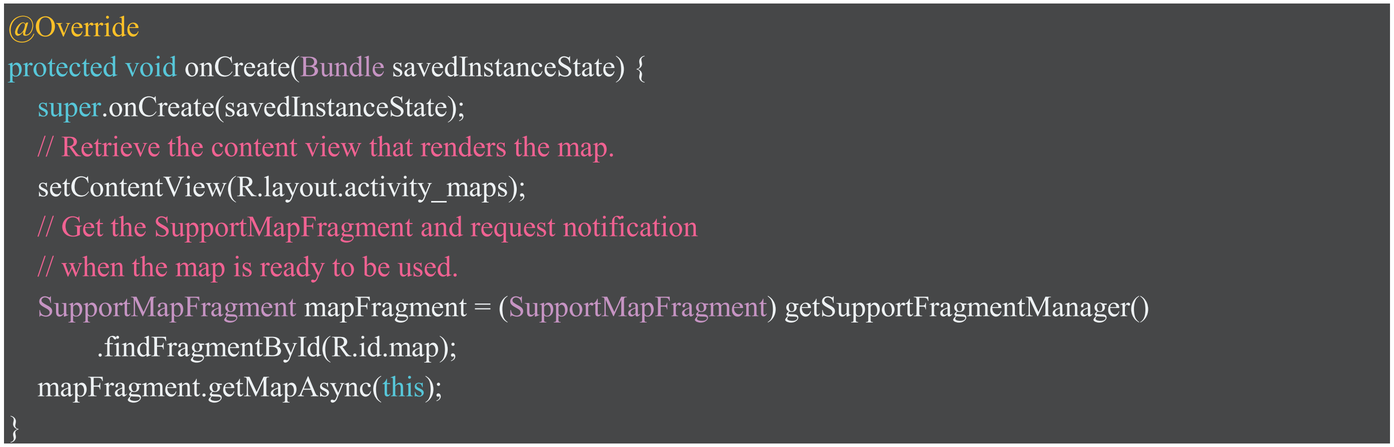
Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

US9467838	Exemplary Supporting Evidence Regarding Accused Products



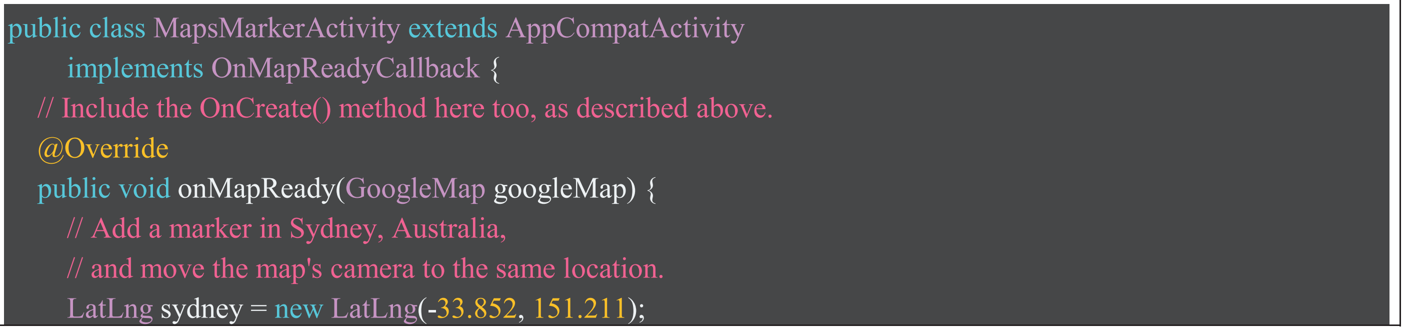
```
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.mapwithmarker.MapsMarkerActivity" />
```

In your activity's onCreate() method, set the layout file as the content view. Get a handle to the map fragment by calling FragmentManager.findFragmentById(). Then use getMapAsync() to register for the map callback:



```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    // Retrieve the content view that renders the map.
    setContentView(R.layout.activity_maps);
    // Get the SupportMapFragment and request notification
    // when the map is ready to be used.
    SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
            .findFragmentById(R.id.map);
    mapFragment.getMapAsync(this);
}
```

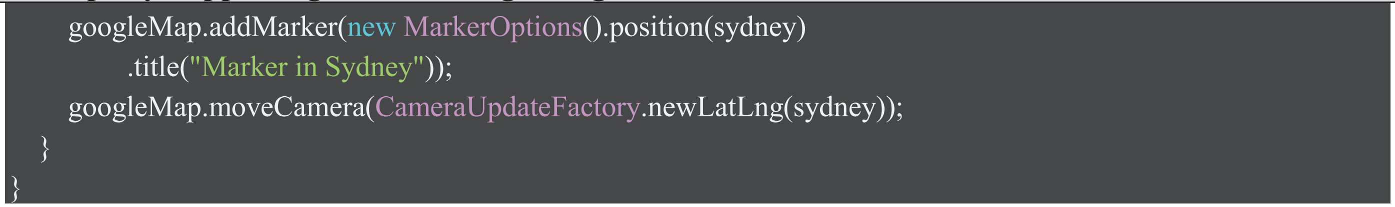
Implement the OnMapReadyCallback interface and override the onMapReady() method, to set up the map when the GoogleMap object is available:



```
public class MapsMarkerActivity extends AppCompatActivity
        implements OnMapReadyCallback {
    // Include the OnCreate() method here too, as described above.
    @Override
    public void onMapReady(GoogleMap googleMap) {
        // Add a marker in Sydney, Australia,
        // and move the map's camera to the same location.
        LatLng sydney = new LatLng(-33.852, 151.211);
```

Exhibit D - Claim Chart for U.S. Patent No. 9,467,838 Against ZTE

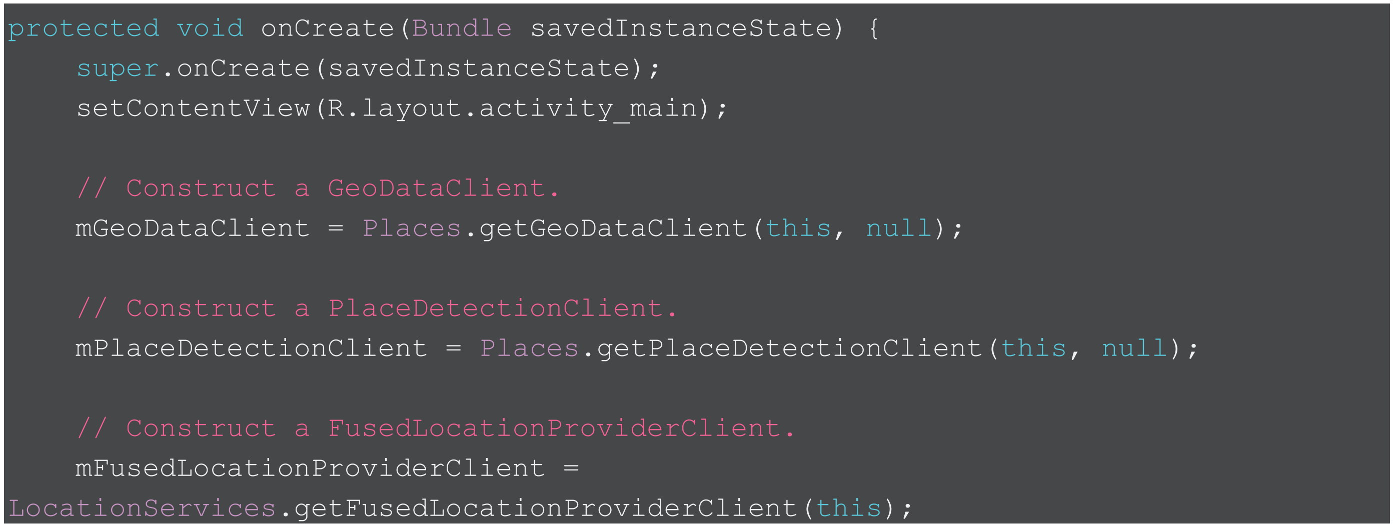
US9467838	Exemplary Supporting Evidence Regarding Accused Products

	
	googleMap.addMarker(new MarkerOptions().position(sydney)
	.title("Marker in Sydney"));
	googleMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));
	}
	}
	See, also, https://developers.google.com/maps/documentation/android-api/marker

[22C] and identifying the particular user-selectable symbol based, at least in part, on the spatial coordinates represented by the selected position.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of identifying the particular user-selectable symbol based, at least in part, on the spatial coordinates represented by the selected position. See, e.g., 1P, 1A, 1G and 1H. Upon information and belief, the first device identifies a symbol based on the determined coordinates. Each marker corresponds to a spatial coordinate (i.e. longitude/latitude) as well as an x/y coordinate on the map display. See, e.g., https://developers.google.com/maps/documentation/android-api/map-with-marker https://developers.google.com/maps/documentation/android-api/marker
23[A]. The method of claim 22, wherein identifying the particular user-selectable symbol based, at least in part, on the spatial coordinates represented by the selected position comprises: searching a database of entities for an entity located nearest to the spatial coordinates represented by the	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 22, wherein identifying the particular user-selectable symbol based, at least in part, on the spatial coordinates represented by the selected position comprises: searching a database of entities for an entity located nearest to the spatial coordinates represented by the selected position, wherein the entities represented by data in the database include the one or more second devices included in the group, wherein the database data include locations of the respective entities, and wherein the database is searchable by location. See, e.g., 1G, 1H, and 22. Upon information and belief, the Accused Products include a database located at the first device or send a search request to a server for performing the searching methods of this limitation. See, e.g.,

US9467838	Exemplary Supporting Evidence Regarding Accused Products
selected position, wherein the entities represented by data in the database include the one or more second devices included in the group, wherein the database data include locations of the respective entities, and wherein the database is searchable by location;	Selection with Markers: https://developers.google.com/maps/documentation/android-api/marker Queries with GeoTagging database: https://developers.google.com/android/reference/com/google/android/gms/location/places/GeoDataClient
[23B] and based on a result of searching the database, identifying the particular second device as the entity located nearest to the spatial coordinates represented by the selected position, wherein the particular user-selectable symbol corresponds to the particular second device.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of, based on a result of searching the database, identifying the particular second device as the entity located nearest to the spatial coordinates represented by the selected position, wherein the particular user-selectable symbol corresponds to the particular second device. See, e.g. 1G and 1H. Upon information and belief, the Accused Devices include a database located either at the first device or send a search request to a server for performing the searching methods of this limitation. See, e.g., Android Maps and Geotagging APIs: Instantiate the Places API clients The following interfaces provide the primary entry points to the Google Places API for Android: • The GeoDataClient provides access to Google's database of local place and business information. • The PlaceDetectionClient provides quick access to the device's current place, and offers the opportunity to report the location of the device at a particular place. The LocationServices interface is the main entry point for Android location services. To use the APIs, instantiate GeoDataClient, PlaceDetectionClient, and FusedLocationProviderClient in your fragment's or activity's onCreate() method, as shown in the following code sample:

<table>
<tr><th>US9467838</th><th>Exemplary Supporting Evidence Regarding Accused Products</th></tr>
<tr><td></td><td>



```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    // Construct a GeoDataClient.
    mGeoDataClient = Places.getGeoDataClient(this, null);

    // Construct a PlaceDetectionClient.
    mPlaceDetectionClient = Places.getPlaceDetectionClient(this, null);

    // Construct a FusedLocationProviderClient.
    mFusedLocationProviderClient =
LocationServices.getFusedLocationProviderClient(this);
```

See, e.g., https://developers.google.com/maps/documentation/android-api/current-place-tutorial
https://developers.google.com/android/reference/com/google/android/gms/location/places/GeoDataClient</td></tr>
<tr><td>24[A]. The method of claim 23, wherein the entity is a first entity, and wherein the method further comprises performing by the first device: receiving user input via user interaction with the interactive display of the first device, the user input</td><td>ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 23, wherein the entity is a first entity, and wherein the method further comprises performing by the first device: receiving user input via user interaction with the interactive display of the first device, the user input specifying a location and a symbol corresponding to a second entity other than the first device and the one or more second devices included in the group. See, e.g., 1G, 1H, 22, and 23. For example, the user of the first device selects a symbol on the map by touching the display. The selection occurs on a touch display of the first device which includes logic for detecting the touch position and corresponding the touch positon to a position on the georeferenced map. See, e.g., 1P an 1A. For example, the user of the first device can select other symbols on the touch display.</td></tr>
</table>

US9467838	Exemplary Supporting Evidence Regarding Accused Products
specifying a location and a symbol corresponding to a second entity other than the first device and the one or more second devices included in the group;	
[24B] and based on the user input, adding the user-specified symbol to the interactive display at a position on the second georeferenced map corresponding to the user-specified location of the second entity.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of, based on the user input, adding the user-specified symbol to the interactive display at a position on the second georeferenced map corresponding to the user-specified location of the second entity. See, e.g., 1G and 1H. For example, the user of the first device selects a symbol on the map by touching the display. The selection occurs on a touch display of the first device which includes logic for detecting the touch position and corresponding the touch positon to a position on the georeferenced map. See, e.g., 1P an 1A. For example, the user of the first device can select other symbols on the touch display and a user can drop a marker on a map and/or share a location by specifying another user/device, specifying a method for sharing, and specifying additional data such as text to be sent with the location/marker.
25. The method of claim 24, further comprising performing by the first device: transmitting the user-specified symbol and location of the second entity to the one or more second devices included in the group for addition of the user-specified symbol to respective interactive displays of the one or more second devices at respective positions on respective georeferenced	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 24, further comprising performing by the first device: transmitting the user-specified symbol and location of the second entity to the one or more second devices included in the group for addition of the user-specified symbol to respective interactive displays of the one or more second devices at respective positions on respective georeferenced maps corresponding to the user-specified location of the second entity. See, e.g., 1P, 1A, 22, 23, and 24. For example, the user of the first device can select other symbols on the touch display and a user can drop a marker on a map and/or share a location by specifying another user/device, specifying a method for sharing, and specifying additional data such as text to be sent with the location/marker. See, e.g., Placing a Marker: https://developers.google.com/maps/documentation/android-api/marker based on queries with GeoTagging database: https://developers.google.com/android/reference/com/google/android/gms/location/places/GeoDataClient

US9467838	Exemplary Supporting Evidence Regarding Accused Products
maps corresponding to the user-specified location of the second entity.	Sharing a link: https://support.google.com/maps/answer/144361?co=GENIE.Platform%3DDesktop&hl=en
26. The method of claim 25, wherein the user input further specifies information associated with the second entity, and wherein the method further comprises performing by the first device: transmitting the user-specified information associated with the second entity to the one or more second devices included in the group.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 25, wherein the user input further specifies information associated with the second entity, and wherein the method further comprises performing by the first device: transmitting the user-specified information associated with the second entity to the one or more second devices included in the group. See, e.g., 1P, 1A, 1B, 22, 23, 24, and 25. For example, the user of the first device can select other symbols on the touch display and a user can drop a marker on a map and/or share a location by specifying another user/device, specifying a method for sharing, and specifying additional data such as text to be sent with the location/marker.
27. The method of claim 26, wherein the information comprises a category of the second entity.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 26, wherein the information comprises a category of the second entity. See, e.g., 1P, 1A, 22, 23, 24, 25, and 26. For example, the user of the first device can select other symbols on the touch display and a user can drop a marker on a map and/or share a location by specifying another user/device, specifying a method for sharing, and specifying additional data such as text to be sent with the location/marker.
28. The method of claim 27, wherein the category comprises a vehicle, a person, an event, a site, a building, or a facility.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 27, wherein the category comprises a vehicle, a person, an event, a site, a building, or a facility. See, e.g., 1P, 1A, 22, 23, 24, 25, and 26. For example, the user of the first device can select other symbols on the touch display and a user can drop a marker on a map and/or share a location by specifying another user/device, specifying a method for sharing, and specifying additional data such as text to be sent with the location/marker. The location can specify a person, an event, or some other location, building, or facility.

US9467838	Exemplary Supporting Evidence Regarding Accused Products
29. The method of claim 26, wherein the information comprises an image.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 26, wherein the information comprises an image. See, e.g., 1P, 1A, 22, 23, 24, 25, and 26. For example, the user of the first device can select other symbols on the touch display and a user can drop a marker on a map and/or share a location by specifying another user/device, specifying a method for sharing, and specifying additional data such as text to be sent with the location/marker. When information is shared by message, the user may also transmit an image.
30. The method of claim 26, wherein the information comprises at least one type of information selected from the group consisting of text and video.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 26, wherein the information comprises at least one type of information selected from the group consisting of text and video. See, e.g., 1P, 1A, 22, 23, 24, 25, and 26. For example, the user of the first device can select other symbols on the touch display and a user can drop a marker on a map and/or share a location by specifying another user/device, specifying a method for sharing, and specifying additional data such as text to be sent with the location/marker. When information is shared by message, the user may also transmit text and/or video.
31. The method of claim 26, further comprising performing by the first device: identifying user interaction with the interactive display selecting the symbol corresponding to the second entity, and based thereon, displaying the information associated with the second entity.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 26, further comprising performing by the first device: identifying user interaction with the interactive display selecting the symbol corresponding to the second entity, and based thereon, displaying the information associated with the second entity. See, e.g., 1P, 1A, 22, 23, 24, 25, and 26. For example, the user of the first device can select other symbols on the touch display and a user can drop a marker on a map and/or share a location by specifying another user/device, specifying a method for sharing, and specifying additional data such as text to be sent with the location/marker.
32. The method of claim 31, wherein the first device uses an Internet Protocol to transmit the user-specified symbol, location, and	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 31, wherein the first device uses an Internet Protocol to transmit the user-specified symbol, location, and information associated with the second entity. See, e.g., 1P, 1A, 22, 23, 24, 25, 26, and 31. For example, the user of the first device can select other symbols on the touch display and a user can drop a marker on a map and/or share a location by specifying another user/device, specifying a method for sharing, and specifying additional data such as text to be sent with the

US9467838	Exemplary Supporting Evidence Regarding Accused Products
information associated with the second entity.	location/marker. See, e.g., 1P and 1A above regarding the hardware and software for transmitting data over the Internet. See, e.g., 1C and 1D regarding the employment of location services and location sharing to obtain and send locations over the Internet. For example, both Find My Friends and Find My Family use location services and location sharing to obtain locations and send those locations to corresponding servers over the Internet for transmission to recipient devices/users. For example, sending messages on the Accused Devices using Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, Google Chrome, and/or associated services includes sending communication over IP.
33. The method of claim 26, further comprising performing by the first device: adding data representing the spatial coordinates of the location of the second entity and data representing the information associated with the second entity to the database.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 26, further comprising performing by the first device: adding data representing the spatial coordinates of the location of the second entity and data representing the information associated with the second entity to the database. See, e.g., 1P, 1A, 22, 23, 24, 25, and 26. For example, the user of the first device can select other symbols on the touch display and a user can drop a marker on a map and/or share a location by specifying another user/device, specifying a method for sharing, and specifying additional data such as text to be sent with the location/marker.
34[A]. The method of claim 24, wherein the portion of the interactive display is a first portion, wherein the position of the symbol corresponding to the particular second device is a first position, and wherein receiving the user input specifying the	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 24, wherein the portion of the interactive display is a first portion, wherein the position of the symbol corresponding to the particular second device is a first position, and wherein receiving the user input specifying the location of the second entity comprises: detecting user selection of a second portion of the interactive display corresponding to a second position on the second georeferenced map. See, e.g., 1G and 1H. For example, the user of the first device selects a symbol on the map by touching the display. The selection occurs on a touch display of the first device which includes logic for detecting the touch position and corresponding the touch positon to a position on the georeferenced map. See, e.g., 1P an 1A.

US9467838	Exemplary Supporting Evidence Regarding Accused Products
location of the second entity comprises: detecting user selection of a second portion of the interactive display corresponding to a second position on the second georeferenced map;	
[34B] and based at least in part on coordinates of the second position on the second georeferenced map and on the second georeferenced map data relating positions on the second georeferenced map to spatial coordinates, determining spatial coordinates of a location represented by the second position on the second georeferenced map, wherein the location represented by the second position is the location of the second entity.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of, based at least in part on coordinates of the second position on the second georeferenced map and on the second georeferenced map data relating positions on the second georeferenced map to spatial coordinates, determining spatial coordinates of a location represented by the second position on the second georeferenced map, wherein the location represented by the second position is the location of the second entity. See, e.g., 1P, 1A, 1G and 1H. Upon information and belief, the first device determines spatial coordinates representing selected positions on the map.
35. The method of claim 23, wherein the database	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 23, wherein the database is stored on the first device. See, e.g., 1P,

US9467838	Exemplary Supporting Evidence Regarding Accused Products
is stored on the first device.	1A, 1G and 1H. Upon information and belief, the first device keeps one or more databases for the purposes of Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome.
36. The method of claim 23, wherein the database is stored on the first server.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 23, wherein the database is stored on the first server. See, e.g., 1P, 1A, 1G and 1H. Upon information and belief, the Accused Products communicate with servers keeping databases for the same purposes discussed in claims 1 and 23, e.g., Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome. See, e.g., 1E and 1F.
37[A]. The method of claim 1, further comprising performing by the first device: receiving user-specified information transmitted by a particular second device, the user-specified information including a user-specified location and a user-specified symbol corresponding to an entity other than the first device and the one or more second devices included in the group;	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, further comprising performing by the first device: receiving user-specified information transmitted by a particular second device, the user-specified information including a user-specified location and a user-specified symbol corresponding to an entity other than the first device and the one or more second devices included in the group. See, e.g., 1P and 1A above regarding the hardware and software for transmitting data over the Internet. See, e.g., 1C and 1D regarding the use of location services and location sharing to obtain and send locations over the Internet. For example, Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome use location services and location sharing to obtain locations and send/receive those locations to corresponding servers over the Internet for transmission to recipient devices/users. For example, receiving messages on the Accused Devices using Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome and associated servers for the same includes sending communication over IP.
[37B] and adding the user-specified symbol to the interactive display at a position on the second	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of adding the user-specified symbol to the interactive display at a position on the second georeferenced map corresponding to the user-specified location. See, e.g., 1P and 1A above regarding the hardware and software for transmitting data over the Internet. See, e.g., 1C and 1D regarding the use of

US9467838	Exemplary Supporting Evidence Regarding Accused Products
georeferenced map corresponding to the user-specified location.	location services and location sharing to obtain and send locations over the Internet. For example, Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome use location services and location sharing to obtain locations and send/receive those locations to corresponding servers over the Internet for transmission to recipient devices/users. For example, receiving messages on the Accused Devices using Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome as associated services includes sending communication over IP. For example, a first device receives shared locations and markers from second devices and displays them on a map on the first device.
38. The method of claim 37, further comprising performing by the first device: identifying user interaction with the interactive display selecting the user-specified symbol corresponding to the entity, and based thereon, displaying information associated with the entity, wherein the user-specified information further includes the information associated with the entity.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 37, further comprising performing by the first device: identifying user interaction with the interactive display selecting the user-specified symbol corresponding to the entity, and based thereon, displaying information associated with the entity, wherein the user-specified information further includes the information associated with the entity. See, e.g., 1P and 1A above regarding the hardware and software for transmitting data over the Internet. See, e.g., 1C and 1D regarding the use of location services and location sharing to obtain and send locations over the Internet. For example, Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome use location services and location sharing to obtain locations and send/receive those locations to corresponding servers over the Internet for transmission to recipient devices/users. For example, receiving messages on the Accused Devices using Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome and associated services includes sending communication over IP. For example, a first device receives shared locations and markers from second devices and displays them on a map on the first device. A user of the first device may then interact with the received shared location and/or marker and the map data and specify additional information.
39. The method of claim 1, wherein the message including the identifier	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, wherein the message including the identifier corresponding to the group is a first message, and wherein the method further comprises performing by the first device:

US9467838	Exemplary Supporting Evidence Regarding Accused Products
corresponding to the group is a first message, and wherein the method further comprises performing by the first device: sending, to a particular second device via the first server, a second message related to remotely controlling the particular second device to perform an action, wherein the particular second device is configured to perform the action based on receiving the second message.	sending, to a particular second device via the first server, a second message related to remotely controlling the particular second device to perform an action, wherein the particular second device is configured to perform the action based on receiving the second message. See, e.g., 1H. The user of the first device sends an action by message, via corresponding servers, to remotely control the second device upon receiving a message from the server corresponding to the action by message.
40. The method of claim 39, wherein the second message indicates the action to be performed, and wherein the action is selected from the group consisting of playing audio, initiating a phone call, vibrating, converting text to speech, changing sound intensity, and displaying information.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 39, wherein the second message indicates the action to be performed, and wherein the action is selected from the group consisting of playing audio, initiating a phone call, vibrating, converting text to speech, changing sound intensity, and displaying information. See, e.g., 1H. For example, a user may play a sound, initiate a call, and display information. Upon information and belief, the second device vibrates, converts text to speech, and changes sound intensity in accordance with the claim.
41. The method of claim	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to

US9467838	Exemplary Supporting Evidence Regarding Accused Products
40, wherein playing audio comprises playing an audio message announcing an emergency.	the performance of the method of claim 40, wherein playing audio comprises playing an audio message announcing an emergency. See, e.g., 1H and 40. For example, a user may play a sound, initiate a call, and display information to announce an emergency, e.g. lost device.
42. The method of claim 1, wherein the message including the identifier corresponding to the group is a first message, and wherein the method further comprises performing by the first device: receiving a second message sent by a particular second device, wherein the second message indicates an action to be performed by the first device; and performing the indicated action.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, wherein the message including the identifier corresponding to the group is a first message, and wherein the method further comprises performing by the first device: receiving a second message sent by a particular second device, wherein the second message indicates an action to be performed by the first device; and performing the indicated action. See, e.g., 1H. The first device, because it is running Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome, receives and performs actions identified in 1H.
43. The method of claim 42, wherein the indicated action is selected from the group consisting of playing audio, initiating a phone call, vibrating, converting text to speech, changing sound intensity, and displaying	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 42, wherein the indicated action is selected from the group consisting of playing audio, initiating a phone call, vibrating, converting text to speech, changing sound intensity, and displaying information. See, e.g., 1H. For example, a user may play a sound, initiate a call, and display information. Upon information and belief, the second device vibrates, converts text to speech, and changes sound intensity in accordance with the claim.

US9467838	Exemplary Supporting Evidence Regarding Accused Products
information.	
44[A]. The method of claim 1, further comprising performing by the first device: presenting another symbol on the second georeferenced map corresponding to a fixed location and associated with a telephone number;	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, further comprising performing by the first device: presenting another symbol on the second georeferenced map corresponding to a fixed location and associated with a telephone number. See, e.g., 1G and 1H. For example, the first device's map can display icons representing fixed entities which correspond to a location and a telephone number.
[44B] and receiving user selection of the other symbol and, based thereon, initiating a telephone call to the telephone number associated with the symbol.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of receiving user selection of the other symbol and, based thereon, initiating a telephone call to the telephone number associated with the symbol. See, e.g., 1G and 1H. For example, the user of the first device can select the icon of a fixed entity to retrieve a telephone number and call the telephone number associated with the fixed entity. See, e.g., 1C and 1D for additional map and location data.
45. The method of claim 1, further comprising performing, by the first device: presenting a symbol corresponding to a facility, wherein the facility is selected from the group consisting of a hospital, a police station, and a fire station, and wherein the symbol corresponding to the	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, further comprising performing, by the first device: presenting a symbol corresponding to a facility, wherein the facility is selected from the group consisting of a hospital, a police station, and a fire station, and wherein the symbol corresponding to the facility is positioned on the second georeferenced map at a position corresponding to a location of the facility. See, e.g. 1H and 44A-B. See, e.g., 1C and 1D for additional map and location data.

US9467838	Exemplary Supporting Evidence Regarding Accused Products
facility is positioned on the second georeferenced map at a position corresponding to a location of the facility.	
46. The method of claim 45, further comprising performing, by the first device: identifying user interaction with the interactive display selecting the symbol corresponding to the facility, and based thereon, displaying information associated with the facility.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 45, further comprising performing, by the first device: identifying user interaction with the interactive display selecting the symbol corresponding to the facility, and based thereon, displaying information associated with the facility. See, e.g., 1G and 1H. For example, the user of the first device can select the icon of a fixed entity to retrieve additional information about the entity, including a telephone number. See, e.g., 1C and 1D for additional map and location data.
47. The method of claim 46, wherein the information associated with the facility comprises a uniform resource locator (URL) of a web site associated with the facility.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 46, wherein the information associated with the facility comprises a uniform resource locator (URL) of a web site associated with the facility. See, e.g., 1G and 1H. For example, the user of the first device can select the icon of a fixed entity to retrieve additional information about the entity, including a website's address. See, e.g., 1C and 1D for additional map and location data.
48. The method of claim 45, further comprising performing, by the first device: identifying user interaction with the interactive display	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 45, further comprising performing, by the first device: identifying user interaction with the interactive display selecting the symbol corresponding to the facility and user interaction with the display specifying an action, and based thereon, loading a web page associated with the facility. See, e.g., 1G and 1H. For example, the user of the first device can select the icon of a fixed entity to retrieve additional information about the entity, including a website's address for selection. See, e.g., 1C

US9467838	Exemplary Supporting Evidence Regarding Accused Products
selecting the symbol corresponding to the facility and user interaction with the display specifying an action, and based thereon, loading a web page associated with the facility.	and 1D for additional map and location data.
49[A]. The method of claim 1, further comprising performing by the first device: identifying user interaction with the interactive display selecting a subset of the user-selectable symbols corresponding to a subset of the one or more second devices included in the group;	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, further comprising performing by the first device: identifying user interaction with the interactive display selecting a subset of the user-selectable symbols corresponding to a subset of the one or more second devices included in the group. See, e.g., 1B. For example, Google Contacts may be grouped by lists. Groups made by linking or associating within Find My Device and Google Maps, taken separately, constitute subsets or subgroups of a group of contacts within the Accused Device. Within the Accused Device, if multiple accounts have been setup for Find My Device, each Account constitutes a subset of second devices. Moreover, groups and subsets may be identified in each of Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome.
[49B] and identifying user interaction with the interactive display specifying an action and, based thereon, assigning the subset of second devices to a sub-net.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the identifying user interaction with the interactive display specifying an action and, based thereon, assigning the subset of second devices to a sub-net. See, e.g., [49A] above.
50[A]. The method of claim 49, further comprising performing	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 49, further comprising performing by the first device: identifying user interaction with the interactive display selecting the sub-net and user interaction with the display

US9467838	Exemplary Supporting Evidence Regarding Accused Products
by the first device: identifying user interaction with the interactive display selecting the sub-net and user interaction with the display specifying an action;	specifying an action. See, e.g., 1C and 49A above. For example, a user of a first device can initiate a call with a member or members of the grouped identified.
[50B] and based thereon, sending fourth data to the subset of second devices via the first server or initiating a phone conference with the subset of second devices.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of, based thereon, sending fourth data to the subset of second devices via the first server or initiating a phone conference with the subset of second devices. See, e.g., 50A above. For example, communication to the recipient devices includes sending data to a server. See, e.g., 1A, 1C, 1D, and 1H above.
51. The method of claim 1, wherein the first server is the second server.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, wherein the first server is the second server. See, e.g., 1A, 1C, 1D, and 1H above, including a server corresponding to Android Device Manager, Find My Phone, Find My Device, Google Latitude, Google Plus, Google Hangouts, Google Maps, Google Assistant, Google Search, Google Messages, Android Messenger, Google Allo, Google Duo, GMail, and Google Chrome and associated services..
52. The method of claim 1, wherein the first set of second devices and the second set of second devices are identical.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, wherein the first set of second devices and the second set of second devices are identical. See, e.g., 1A, 1F, and 1G, where the first and second sets of second devices can be the same.
53. The method of claim 1, wherein the message further includes an identifier corresponding to the first device.	ZTE infringes directly and/or indirectly by performing, inducing others to perform, and/or contributing to the performance of the method of claim 1, wherein the message further includes an identifier corresponding to the first device. See, e.g., 1A and 1B, where the message includes the identifier. Upon information, additional metadata associated with the Accused Device is included within the message.

US9467838	Exemplary Supporting Evidence Regarding Accused Products
54[P]. A system comprising:	ZTE directly and/or indirectly infringes by providing a system [comprising]. See 1P above.
[54A] a first device programmed to perform operations comprising:	ZTE directly and/or indirectly infringes by a first device programmed to perform operations [of claim 54]. See 1A above.
[54B] joining a communication network corresponding to a group, wherein joining the communication network comprises transmitting a message including an identifier corresponding to the group;	ZTE directly and/or indirectly infringes by joining a communication network corresponding to a group, wherein joining the communication network comprises transmitting a message including an identifier corresponding to the group. See 1B above.
[54C] participating in the group, wherein participating in the group includes sending first location information to a first server and receiving second location information from the first server, the first location information comprising a location of the first device, the second location information comprising	ZTE directly and/or indirectly infringes by: participating in the group, wherein participating in the group includes sending first location information to a first server and receiving second location information from the first server, the first location information comprising a location of the first device, the second location information comprising one or more locations of one or more respective second devices included in the group. See 1C above.

US9467838	Exemplary Supporting Evidence Regarding Accused Products
one or more locations of one or more respective second devices included in the group;	
[54D] presenting, via an interactive display of the first device, a first interactive, georeferenced map and a first set of one or more user-selectable symbols corresponding to a first set of one or more of the second devices, wherein the first set of symbols are positioned on the first georeferenced map at respective positions corresponding to the locations of the first set of second devices, and wherein first georeferenced map data relate positions on the first georeferenced map to spatial coordinates;	ZTE infringes directly and/or indirectly by presenting, via an interactive display of the first device, a first interactive, georeferenced map and a first set of one or more user-selectable symbols corresponding to a first set of one or more of the second devices, wherein the first set of symbols are positioned on the first georeferenced map at respective positions corresponding to the locations of the first set of second devices, and wherein first georeferenced map data relate positions on the first georeferenced map to spatial coordinates. See 1D above.
[54E] sending, to a second server, a request for second georeferenced map data different from the first georeferenced map data;	ZTE infringes directly and/or indirectly by: sending, to a second server, a request for second georeferenced map data different from the first georeferenced map data. See 1E above.

US9467838	Exemplary Supporting Evidence Regarding Accused Products
[54F] receiving, from the second server, the second georeferenced map data;	ZTE infringes directly and/or indirectly by receiving, from the second server, the second georeferenced map data. See 1F above.
[54G] presenting, via the interactive display of the first device, a second georeferenced map and a second set of one or more user-selectable symbols corresponding to a second set of one or more of the second devices, wherein the second set of symbols are positioned on the second georeferenced map at respective positions corresponding to the locations of the second set of second devices, and wherein the second georeferenced map data relate positions on the second georeferenced map to spatial coordinates;	ZTE infringes directly and/or indirectly by presenting, via the interactive display of the first device, a second georeferenced map and a second set of one or more user-selectable symbols corresponding to a second set of one or more of the second devices, wherein the second set of symbols are positioned on the second georeferenced map at respective positions corresponding to the locations of the second set of second devices, and wherein the second georeferenced map data relate positions on the second georeferenced map to spatial coordinates. See 1G above.
[54H] and identifying user interaction with the interactive display selecting one or more of the second set of user-	ZTE infringes directly and/or indirectly by identifying user interaction with the interactive display selecting one or more of the second set of user-selectable symbols corresponding to one or more of the second devices and positioned on the second georeferenced map and user interaction with the display specifying an action and, based thereon, sending third data to the selected one or more second devices via the first server.

US9467838	Exemplary Supporting Evidence Regarding Accused Products
selectable symbols corresponding to one or more of the second devices and positioned on the second georeferenced map and user interaction with the display specifying an action and, based thereon, sending third data to the selected one or more second devices via the first server.	See 1H above.