EXHIBIT 23

Exhibit E – U.S. Patent No. 6,424,354

Motorola directly and/or indirectly infringes at least claims 1 and 41 of the '354 patent, either literally or through the doctrine of equivalents. Motorola's infringing products include mobile devices such as smartphones and tablet computers, including but not limited to the: Atrix, Bravo, Cliq, Cliq XT, Cliq 2, Charm, Defy, Devour, BackFlip, Devour, Droid, Droid 2, Droid 2 Global, Droid X, Droid Pro, Flipout, Flipside, i1, Xoom, (collectively, "the '354 Accused Products"). [1]

For the purposes of this analysis, Plaintiffs will examine a representative mobile device, Motorola's Droid X, which is shipped operates with the Android 2.1 Platform. All other '354 Accused Products meet the limitations of the asserted claims on the same bases as indicated for the Droid X, unless otherwise stated.

In addition to Motorola's direct infringement of the claims of the '354 patent through its development, testing, manufacture and use of its devices, Motorola also indirectly infringes claims 1 and 41 of the '354 patent. Manufacturers, retailers, distributors, end-users and others in the distribution channel of the '354 Accused Products directly infringe these claims by using, selling, offering for sale, and/or importing these devices into the United States. Motorola contributes to and induces the infringement of asserted claims 1 and 41 through its promotion and provision of intentional marketing, sale and/or technical support of the '354 Accused Products and associated specialized components in the United States, and through the intentional design, marketing, manufacture, sale, and/or technical support of the '354 Accused Products abroad to induce direct infringement in the United States. Motorola supplies '354 Accused Products and actively encourages the use, sale, offer for sale, and importation of the same in the United States through the promotion and provision of marketing literature and user guides, which induces and results in direct infringement. *See, e.g.*, Motorola Droid X User Guide (WI-Apple0034078-34145). Upon information and belief, Motorola has known or should have known that these actions would cause direct infringement of the '354 patent and did so with specific intent to encourage direct infringement. Additionally, the '354 Accused Products have no substantial non-infringing uses.

These infringement contentions are preliminary and based only on publicly available information as to the '354 Accused Products. Motorola has not yet provided discovery as to its accused products and in addition Plaintiff's investigation of Motorola's infringement is ongoing. Based on discovery and Plaintiff's continued investigations Plaintiff reserves the right to amend these contentions to identify additional bases for infringement and additional accused products, including products that Motorola may introduce in the future. Accordingly, Plaintiff reserves its right to amend these contentions as discovery and its investigation proceeds. Also, these disclosures are made based on information ascertained to date, and Plaintiff expressly reserves the right to

---

[1] Motorola has announced additional smartphones including XRT and Titanium which may also infringe the '354 Patent. Apple reserves the right to supplement this analysis and this list of accused products as discovery into these newly announced products progresses.

modify or amend the disclosures contained herein based on the Court's claim constructions or to reflect additional information that becomes available to Plaintiff.

U.S. Patent No. 6,424,354	Infringement Contentions
1. A method for operating a computer-implemented event notification system for propagating, among a plurality of objects, events representing changes in the objects, the operating method comprising the steps of:	The '354 Accused Products perform a method for operating a computer-implemented event notification system for propagating, among a plurality of objects, events representing changes in the objects. • For example, the Motorola Droid X includes a Texas Instruments OMAP3630 processor for executing applications such as web browsers, email clients, and telephony applications, and is therefore a computer. *See* **Exh. E-8** [Motorola Droid X Specification]. • For example, Android uses objects called "Intents" to broadcast event notifications: "[a]n intent is an abstract description of an operation to be performed. It can be used with . . . broadcastIntent to send it to any interested BroadcastReceiver components." *See* **Exh. E-1** [Android Dev Site, "Intent"]. Interested objects can register to receive broadcasts of intents that they are interested in: the Context.registerReceiver() method will "[r]egister a BroadcastReceiver to be run in the main activity thread. The *receiver* will be called with any broadcast Intent that matches *filter*, in the main application thread." *See* **Exh. E-2** [Android Dev Site, "Context"]. An event producer can then use the Context.sendBroadcast() method to "[b]roadcast [a] given intent to all interested BroadcastReceivers." *Id.* The events propagated by the '354 Accused Products include, among other things, events representing changes in objects. • For example, the '354 Accused Products include a Settings application that includes a "Wireless controls" screen that includes a checkbox user interface element that controls whether the phone is in "Airplane mode," in which all of its wireless transceivers, including its Bluetooth transceiver, are disabled.

U.S. Patent No. 6,424,354	Infringement Contentions
	 Fig. 1, Motorola Droid X with Airplane mode disabled. • For example, the ’354 Accused Products further include an AirplaneModeEnabler object that tracks the Airplane mode checkbox state. Among other things, this object includes the setAirplaneModeOn() method. In this method, the state of the system object is changed using the Settings.System.putInt() method, so that this object includes the AIRPLANE_MODE_ON state. The setAirplaneModeOn() method then use the Context.sendBroadcast() method to send an event representing the change in the Airplane Mode state that was

U.S. Patent No. 6,424,354	Infringement Contentions
	reflected in the system object. *See* **Exh. E-3** [AirplaneModeEnabler.java].
(a) creating, on behalf of a first object, connection information representing the first object's interest in, and an associated object method for, receiving notification of a change to a second object;	The '354 Accused Products perform the step of creating, on behalf of a first object, connection information representing the object's interest in, and an associated object method for, receiving a notification. • For example, the Context.registerReceiver() method allows an object to "[r]egister to receive intent broadcasts, to run in the main activity thread." *See* **Exh. E-2** [Android Dev Site, "Context"]. This method takes two arguments, called "receiver" and "filter." The receiver argument points to an object method to be called to receive notification of a change to a second object: "The *receiver* will be called with any broadcast intent that matches *filter*, in the main application thread." *Id*. The *filter* parameter that is passed to registerReceiver() is an object of type IntentFilter, and "[s]elects the Intent broadcasts to be received." *Id.* The IntentFilter includes a "[s]tructured description of values to be matched. An IntentFilter can match against actions, categories, and data (either via its type, scheme, and/or path) in an Intent. It also includes a "priority" value which is used to order multiple matching filters." *See* **Exh. E-4** [Android Dev Site, "IntentFilter"]. • For example, in the '354 Accused Products, objects instantiated from the class BluetoothService constitute part of the software interface to the Bluetooth hardware device; methods in this class "[b]ring down bluetooth" and "tur[n] on/off the underlying hardware." *See* **Exh. E-5** [BluetoothService.java]. These objects contain an initialization method, init(), which "[m]ust be called after construction, and before any other method." This init() method calls the BluetoothService method registerForAirplaneMode(), which registers a receiver called mReceiver that includes an IntentFilter that is designed to receive events of type ACTION_AIRPLANE_MODE_CHANGED. *Id*. The mReceiver object is an instance of BroadcastReceiver instantiated by BluetoothService. *Id.* In addition, mReceiver constitutes an object method for receiving notification of the registered events. *Id.* The intent ACTION_AIRPLANE_MODE_CHANGED is defined in the class android.content.Intent. *See*, **Exh. E-6** [Intent.java]. The call to registerReceiver above includes, via the IntentFilter object, information representing the BluetoothService's interest in events of type

U.S. Patent No. 6,424,354	Infringement Contentions
	ACTION_AIRPLANE_MODE_CHANGED. The call further includes, in the form of the mReceiver object, information representing an object method for receiving information of events of type ACTION_AIRPLANE_MODE_CHANGED. The object method of mReceiver which receives the notification is called onReceive(), inherited from BroadcastReceiver, which notes that "[t]his method [onReceive()] is called when the BroadcastReceiver is receiving an Intent broadcast." *See* **Exh. E-7** [Android Dev Site, "BroadcastReceiver"].
(b) registering the connection information with a connection object;	The '354 Accused Products perform the step of registering the connection information with a connection object. • For example, Android's BroadcastReceiver class is the "[b]ase class for code that will receive intents sent by sendBroadcast(). You can either dynamically register an instance of this class with Context.registerReceiver() or statically publish an implementation through the <receiver> tag in your AndroidManifest.xml." *See* **Exh. E-7** [Android Dev Site, "BroadcastReceiver"]. The registerReceiver() method is a method associated with the Context object, with which the relevant connection information is registered. The Context.registerReceiver() method, which "[r]egister[s] a BroadcastReceiver to be run in the main activity thread," registers connection information with the Context object. *See* **Exh. E-2** [Android Dev Site, "Context"]. This information includes, among other things, a BroadcastReceiver object and an IntentFilter object. *Id.* The BroadcastReceiver object includes "code that will receive intents sent by sendBroadcast()" *See* **Exh. E-7** [Android Dev Site, "BroadcastReceiver"]. The IntentFilter object includes a "[s]tructured description of Intent values to be matched" to filter out only those events the BroadcastReceiver wishes to receive. *See* **Exh. E-4** [Android Dev Site, "IntentFilter"]. • For example, as described above the BluetoothService object within the '354 Accused Products uses the Context.registerReceiver() method to register connection information regarding its interest in the ACTION_AIRPLANE_MODE_CHANGED intent with an appropriate Context

U.S. Patent No. 6,424,354	Infringement Contentions
	object.
(c) creating an event representing a change in the second object, responsive to the change in the second object; and	The ’354 Accused Products perform the step of creating an event representing a change in the second object, responsive to the change in the second object. • For example, an object that produces a notification may call the Context.sendBroadcast() method to “[b]roadcast the given intent to all interested BroadcastReceivers. This call is asynchronous; it returns immediately, and you will continue executing while the receivers are run.” *See* **Exh. E-2** [Android Dev Site, “Context”]. • For example, when the Airplane mode checkbox in the “Wireless controls” screen of the Settings application is checked (changed to the on state), the state of the system settings object is changed to AIRPLANE_MODE_ON, and an ACTION_AIRPLANE_MODE_CHANGED Intent is created. *See* **Exh. E-3** [AirplaneModeEnabler.java].
(d) notifying the first object of the event by invoking the associated object method for receiving notification registered with the connection object only if the event information corresponds to an interest registered on behalf of the first object.	The ’354 Accused Products perform the step of notifying the first object of an event by invoking the associated object method for receiving notification registered with the connection object only if the event information corresponds to an interest registered on behalf of the first object. • For example, the Context.registerReceiver() method “[r]egister[s] a BroadcastReceiver to be run in the main activity thread. The *receiver* will be called with any broadcast Intent that matches *filter*, in the main application thread.” *See* **Exh. E-2** [Android Dev Site, “Context”]. Thus, when a broadcast Intent matches the filter settings registered by a receiver, the object method onReceive() of the receiver that has been registered with the Context object using the registerReceiver() method will be called. • For example, when the action ACTION_AIRPLANE_MODE_CHANGED intent is broadcast, the onReceive() method of the mReceiver that was registered with the context object using the registerForAirplaneMode() method in the BluetoothService class will be invoked.
41. A method for operating a computer-	The ’354 Accused Products perform a method for operating a computer-implemented

U.S. Patent No. 6,424,354	Infringement Contentions
implemented event notification system for propagating, among a plurality of objects, events representing changes in the objects, the operating method comprising the steps of:	event notification system for propagating, among a plurality of objects, events representing changes in the objects. • For example, the Motorola Droid X includes a Texas Instruments OMAP3630 processor for executing applications such as web browsers, email clients, and telephony applications, and is therefore a computer. *See* **Exh. E-8** [Motorola Droid X Specification]. • For example, Android uses objects called "Intents" to broadcast event notifications: "[a]n intent is an abstract description of an operation to be performed. It can be used with . . . broadcastIntent to send it to any interested BroadcastReceiver components." *See* **Exh. E-1** [Android Dev Site, "Intent"]. Interested objects can register to receive broadcasts of intents that they are interested in: the registerReceiver method will "[r]egister a BroadcastReceiver to be run in the main activity thread. The *receiver* will be called with any broadcast Intent that matches *filter*, in the main application thread." *See* **Exh. E-2** [Android Dev Site, "Context"]. An event producer can then use the sendBroadcast method to "[b]roadcast [a] given intent to all interested BroadcastReceivers." *Id.* The events propagated by the '354 Accused Products include, among other things, events representing changes in objects. • For example, the '354 Accused Products include a Settings application that includes a "Wireless controls" screen that includes a checkbox user interface element that controls whether the phone is in "Airplane mode," in which all of its wireless transceivers, including its Bluetooth transceiver, are disabled.

U.S. Patent No. 6,424,354	Infringement Contentions
	 Fig. 1, Motorola Droid X with Airplane mode disabled. • For example, the '354 Accused Products further include an AirplaneModeEnabler object that tracks the Airplane mode checkbox state. Among other things, this object includes the setAirplaneModeOn() method. In this method, the state of the a system object is changed using the Settings.System.putInt() method, so that this object now includes the AIRPLANE_MODE_ON state. The setAirplaneModeOn() method then use the sendBroadcast method to send an event representing the change in the Airplane Mode state that was reflected in the

U.S. Patent No. 6,424,354	Infringement Contentions
	system object. *See* **Exh. E-3** [AirplaneModeEnabler.java].
(a) creating, on behalf of a receiver object, connection information representing the receiver object's interest in, and an associated object method for, receiving notification of a change to a source object:	The '354 Accused Products perform the step of creating, on behalf of a receiver object, connection information representing the receiver object's interest in, and an associated object method for, receiving a notification regarding a change in a source object. • For example, the registerReceiver method allows an object to "[r]egister to receive intent broadcasts, to run in the main activity thread." *See* **Exh. E-2** [Android Dev Site, "Context"]. This method takes two arguments, called "receiver" and "filter." The receiver argument points to an object method to be called to receive notification of a change to a second object: "The *receiver* will be called with any broadcast intent that matches *filter*, in the main application thread." *Id.* The *filter* parameter that is passed to RegisterReceiver is an object of type IntentFilter, and "[s]elects the Intent broadcasts to be received." *Id.* The IntentFilter includes a "[s]tructured description of values to be matched. An IntentFilter can match against actions, categories, and data (either via its type, scheme, and/or path) in an Intent. It also includes a "priority" value which is used to order multiple matching filters." *See* **Exh. E-4** [Android Dev Site, "IntentFilter"]. • For example, in the '354 Accused Products, objects instantiated from the class BluetoothService constitute part of the software interface to the Bluetooth hardware device; methods in this class "[b]ring down bluetooth" and "tur[n] on/off the underlying hardware." *See* **Exh. E-5** [BluetoothService.java]. These objects contain an initialization method, init(), which "[m]ust be called after construction, and before any other method." This init() method calls the BluetoothService method registerForAirplaneMode(), which registers a receiver called mReceiver that includes an IntentFilter that is designed to receive events of type ACTION_AIRPLANE_MODE_CHANGED. *Id.* The mReceiver object is an instance of BroadcastReceiver instantiated by BluetoothService. *Id.* In addition, mReceiver constitutes an object method for receiving notification of the registered events. *Id.* The intent ACTION_AIRPLANE_MODE_CHANGED is defined in the class android.content.Intent. *See*, **Exh. E-6** [Intent.java]. The call to registerReceiver above includes, via the IntentFilter object, information

U.S. Patent No. 6,424,354	Infringement Contentions
	representing the BluetoothService's interest in events of type ACTION_AIRPLANE_MODE_CHANGED. The call further includes, in the form of the mReceiver object, information representing an object method for receiving information of events of type ACTION_AIRPLANE_MODE_CHANGED. The object method of mReceiver which receives the notification is called onReceive(), inherited from BroadcastReceiver, which notes that "[t]his method [onReceive()] is called when the BroadcastReceiver is receiving an Intent broadcast." *See* **Exh. E-7** [Android Dev Site, "BroadcastReceiver"].
(b) registering the connection information using a connection object;	The '354 Accused Products perform the step of registering the connection information using a connection object. • For example, Android's BroadcastReceiver class is the "[b]ase class for code that will receive intents sent by sendBroadcast(). You can either dynamically register an instance of this class with Context.registerReceiver() or statically publish an implementation through the <receiver> tag in your AndroidManifest.xml." *See* **Exh. E-7** [Android Dev Site, "BroadcastReceiver"]. The registerReceiver() method is a method associated with the Context object, with which the relevant connection information is registered. The registerReceiver() method, which "[r]egister[s] a BroadcastReceiver to be run in the main activity thread," registers connection information with the Context object. *See* **Exh. E-2** [Android Dev Site, "Context"]. This information includes, among other things, a BroadcastReceiver object and an IntentFilter object. *Id.* The BroadcastReceiver object includes "code that will receive intents sent by sendBroadcast()" *See* **Exh. E-7** [Android Dev Site, "BroadcastReceiver"]. The IntentFilter object includes a "[s]tructured description of Intent values to be matched" to filter out only those events the BroadcastReceiver wishes to receive. *See* **Exh. E-4** [Android Dev Site, "IntentFilter"]. • For example, as described above the BluetoothService object within the '354 Accused Products uses the registerReceiver() method to register connection information regarding its interest in the

U.S. Patent No. 6,424,354	Infringement Contentions
	ACTION_AIRPLANE_MODE_CHANGED intent with an appropriate Context object.
(c) creating an event representing a change in the source object, responsive to the change in the source object;	The '354 Accused Products perform the step of creating an event representing a change in the source object, responsive to the change in the source object. • For example, an object that produces a notification may call the sendBroadcast() method to "[b]roadcast the given intent to all interested BroadcastReceivers. This call is asynchronous; it returns immediately, and you will continue executing while the receivers are run." *See* **Exh. E-2** [Android Dev Site, "Context"]. • For example, when the Airplane mode checkbox in the "Wireless controls" screen of the Settings application is checked (changed to the on state), the state of the system settings object is changed to AIRPLANE_MODE_ON, and an ACTION_AIRPLANE_MODE_CHANGED Intent is created. *See* **Exh. E-3** [AirplaneModeEnabler.java].
(d) notifying the receiver object of the event by invoking the associated object method for receiving notification registered using the connection object only if the event information corresponds to an interest registered on behalf of the receiver object; and	The '354 Accused Products perform the step of notifying the receiver object of an event by invoking the associated object method for receiving notification registered using the connection object only if the event information corresponds to an interest registered on behalf of the receiver object. • For example, the registerReceiver() method "[r]egister[s] a BroadcastReceiver to be run in the main activity thread. The *receiver* will be called with any broadcast Intent that matches *filter*, in the main application thread." *See* **Exh. E-2** [Android Dev Site, "Context"]. Thus, when a broadcast Intent matches the filter settings registered by a receiver, the object method onReceive() of the receiver that has been registered with the Context object using the registerReceiver() method will be called. • For example, when the action ACTION_AIRPLANE_MODE_CHANGED intent is broadcast, the onReceive() method of the mReceiver that was registered with the context object using the registerForAirplaneMode() method in the BluetoothService class will be invoked.

U.S. Patent No. 6,424,354	Infringement Contentions
(e) using the connection information in the connection object to configure status information to enable the notifying step (d).	The ’354 Accused Products perform the step of using the connection information in the connection object to configure status information to enable the notifying step (d). • For example, the ’354 Accused Products can use information regarding a receiver stored in their Context objects to determine whether the receiver has the appropriate permission status to allow notification to occur. The method sendBroadcast(Intent intent, String ReceiverPermission) is used to “[b]roadcast the given Intent to all interested BroadcastReceivers, allowing an optional permission to be enforced.” In particular, the ReceiverPermission argument is used to “nam[e] permissions that a receiver must hold in order to receive your broadcast.” *See* **Exh. E-2** [Android Dev Site, “Context”]. If a given receiver does have the required information, notifications to it will be enabled.