

Apple v. Samsung
Confidential – Attorneys’ Eyes Only

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

UNITED STATES DISTRICT COURT
NORTHERN DISTRICT OF CALIFORNIA
SAN JOSE DIVISION

APPLE INC., a California corporation,

Plaintiff,

v.

SAMSUNG ELECTRONICS CO., LTD., A
Korean business entity; SAMSUNG
ELECTRONICS AMERICA, INC., a New York
corporation; SAMSUNG
TELECOMMUNICATIONS AMERICA, LLC, a
Delaware limited liability company,

Defendants.

Case No. 11-cv-01846-LHK

**DECLARATION OF DR. KARAN
SINGH, PH.D. IN SUPPORT OF
APPLE’S OPPOSITION TO
SAMSUNG’S MOTION FOR
SUMMARY JUDGMENT**

****CONFIDENTIAL – CONTAINS MATERIAL DESIGNATED AS HIGHLY
CONFIDENTIAL – ATTORNEYS’ EYES ONLY PURSUANT
TO A PROTECTIVE ORDER****

SUBMITTED UNDER SEAL

1 I, KARAN SINGH, do hereby declare as follows:

2 1. I have personal knowledge of, and am competent to testify to, the facts and
3 opinions set forth herein.

4 2. I have been asked by counsel for Apple to provide an expert declaration in the
5 above-captioned case. I submit this Declaration in support of Apple's Opposition to Samsung's
6 Motion for Summary Judgment. I have been asked by counsel to review and respond to the
7 opinions and assertions made in the Declaration of Stephen Gray in Support of Samsung's
8 Motion for Summary Judgment.

9 3. I reserve the right to supplement this Declaration if additional data or other
10 information that affects my opinions becomes available or to respond to any additional matters
11 that may addressed by any witness testifying on behalf of Samsung, if asked to do so.

12 4. I am being compensated at my standard hourly consulting rate of \$450 and my
13 compensation is in no way dependent upon the opinions I offer or upon the outcome of the
14 litigation between Apple and Samsung.

15 **II. QUALIFICATIONS**

16 5. I am the same Dr. Karan Singh who submitted an opening expert report regarding
17 Samsung's infringement of U.S. Patent Nos. 7,864,163 (the "'163 patent"), 7,844,915 (the "'915
18 patent") and 7,853,891 ("Infringement Report"), and a rebuttal expert report regarding the
19 validity of those patents ("Validity Report"). The portions of my Infringement Report relating to
20 the '915 patent are attached hereto as Exhibit 1. The portions of my Validity Report relating to
21 the '915 patent and the '163 patent, including the LaunchTile reference, are attached hereto as
22 Exhibit 2.

23 6. Here, I provide a brief summary of my qualifications. My qualifications and
24 experience are stated more fully in my curriculum vitae, which includes a list of all my honours,
25 patents, presentations, grants, and publications from the last five years, and is attached to this
26 Declaration as Exhibit 3.

27 7. I received my Bachelor of Technology degree in Computer Science from the
28 Indian Institute of Technology in 1991. I was awarded a Master of Science degree in 1992, and a

1 Ph.D. in 1995, both in Computer and Information Science, from Ohio State University. I can read
2 and program fluently in object-oriented programming languages, such as C++ and Java.

3 8. In 1994, I was invited to conduct research at the Advanced Telecommunications
4 Research laboratory in Kyoto, Japan. During this time I researched virtual reality technology,
5 specifically designing graphical environments in which human characters could interact with
6 computing systems.

7 9. My Ph.D. dissertation, which I presented in 1995, was on creating representations
8 of humans which could interact in graphical environments.

9 10. In 1995, I joined Alias Wavefront in Toronto, Canada. While there I designed
10 character animation and facial modeling tools for the first release of Maya, which is a software
11 system for computer graphical modeling, animation, and rendering which won a technical Oscar
12 in 2003, one of only 38 such awards since 1930. This software, which I worked on for more than
13 two years, is still the premiere software package today for these functions. I worked at Alias
14 Wavefront until 1999.

15 11. I have worked with Chris Landreth, a director of animated films, since I started
16 with Alias Wavefront in 1995. Chris and I worked together on the design of Maya, and have
17 subsequently worked on a number of film projects. Notable among these projects is the short film
18 "Ryan," which won an Oscar for Best Animated Short in 2005.

19 12. Later in 1999, I joined a start-up company in California called Paraform Inc.
20 While there I worked to develop a system which transformed data from real objects which had
21 been scanned using lasers into useable digital models for downstream applications.

22 13. For several months in 1999 I was a Visiting Professor of Computer Science at the
23 University of Otago in New Zealand. During that time I taught and conducted research in
24 computer graphics.

25 14. Since 2002, I have been an Associate Professor of Computer Science at the
26 University of Toronto where I co-direct a graphics and human computer interaction laboratory
27 known as dgp (dynamic graphics project). I have conducted research and taught classes in
28 graphics and in human computer interaction. During this period, I have also undertaken

1 consulting projects with various companies in the computer graphics and design industries. Since
2 2002, I have also been the Chief Scientist at Geometry Systems, which is a company which
3 designs software for the reverse engineering of physical objects into usable digital models. I also
4 co-founded Arcestra, Inc. in 2006, which is a software service for conceptualizing and visualizing
5 architectural interiors.

6 15. My current research focus is on interaction techniques for pen and touch based
7 devices inspired by a sketching metaphor.

8 16. I have previously testified by deposition as an expert in proceedings before the
9 International Trade Commission in the ITC Investigation In re Certain Electronic Digital Media
10 Devices and Components Thereof, Inv. No. 337-TA-796 on behalf of complainant Apple.

11 **III. MATERIALS REVIEWED**

12 17. In forming my opinions as stated in this Declaration I reviewed a large volume of
13 materials relating to the '915 patent and the '163 patent and to Samsung's arguments with respect
14 to those patents. A subset of the materials I reviewed included the '163 and '915 patents and their
15 file histories, the LaunchTile videos and papers attached to Mr. Bederson's Declaration, the
16 XNav code produced in this litigation, Mr. Gray's Invalidity Report and his Rebuttal Report on
17 Non-Infringement, the Gray and Bederson Declarations and exhibits filed in support of
18 Samsung's Motion for Summary Judgment, and all of the deposition transcript and dictionary
19 excerpts and other materials cited in this Declaration. Additional materials that I reviewed in
20 forming my opinion in this case were disclosed in my opening Infringement Report and in my
21 rebuttal Validity Report.

22 **IV. UNDERSTANDING OF THE LAW**

23
24 18. I have not been asked to offer an opinion on the law; however, as an expert
25 assisting the Court in determining patent infringement and validity, I understand that I am obliged
26 to follow existing law. I have therefore been asked to apply the following legal principles to my
27 analysis of patent infringement and validity:
28

1 19. I understand that to determine whether there is infringement of a patent: (1) the
2 claims of the patent must be construed; and (2) the properly construed claims must then be
3 compared with the accused products.

4 20. Where the Court has construed a claim term, I have applied that construction.
5 Where no claim construction has been issued by the Court, I have interpreted the claims as one of
6 ordinary skill in the art would have at the time the relevant patent was filed in light of its claim
7 language, specification, and prosecution history.

8 21. As the second step in the infringement analysis, I understand that the properly
9 construed claim must be compared to the accused products. I understand that infringement
10 requires that every limitation of a claim be met, either literally or equivalently, by the accused
11 device.

12 22. I understand that one test for determining equivalence is to determine whether the
13 differences between the claimed limitation and the accused product are insubstantial. I
14 understand that another test for determining equivalence is to examine whether the step used by
15 the accused product performs substantially the same function in substantially the same way to
16 achieve substantially the same result as the claimed step.

17 23. I understand that to prove direct infringement of a device claim, a plaintiff must
18 show that a defendant “makes, uses, offers to sell, or sells,” within the United States, or imports
19 into the United States, an accused device that reads on every limitation of the patent claim.

20 24. I understand that a device literally and directly infringes a claim of a patent if all of
21 the asserted claim elements are found in the accused device or method.

22 25. I have been informed by counsel that by United States statute, a patent is presumed
23 valid. I understand that the patent challenger bears the burden of proving invalidity of the patent
24 by clear and convincing evidence.

25 26. I have been informed by counsel that, for a finding of invalidity of a patent under
26 35 U.S.C. § 102, which is known as “anticipation,” each and every element of a claim, as
27 properly construed, must be found either explicitly or inherently in a single prior art reference,
28 subject to the limitations imposed by § 102 in paragraphs (a)–(g). I understand that under

1 principles of inherency, if the extrinsic evidence makes clear that the prior art necessarily
2 functions in accordance with or includes the claimed limitations, it anticipates. I understand that
3 inherency may not be established by probabilities or possibilities. I also understand that, in order
4 to anticipate a patent claim, a prior art reference must also be enabling, such that a person of
5 ordinary skill in the art could practice the invention without undue experimentation.

6 27. I have been informed by counsel that a claim is invalid under 35 U.S.C. § 102(a) if
7 the claimed invention was known or used by others in the U.S., or was patented or published
8 anywhere, before the applicant's invention. I further have been informed that a claim is invalid
9 under 35 U.S.C. § 102(b) if the invention was patented or published anywhere, or was in public
10 use, on sale, or offered for sale in the United States, more than one year prior to the filing date of
11 the patent application. And I have been informed that a patent claim is invalid under 35 U.S.C. §
12 102(e), if an invention described by that claim was described in a U.S. patent granted on an
13 application for a patent by another that was filed in the U.S. before the date of invention for such
14 a claim. A claim is also invalid, as I understand, under 35 U.S.C. §102 (f) if the invention was
15 invented by another prior to the claimed invention. It is also my understanding that a claim is
16 invalid under 35 U.S.C. §102 (g)(2) if, prior to the date of invention for the claim, the invention
17 was made in the U.S. by another who had not abandoned, suppressed or concealed the invention.

18
19 **V. THE SAMSUNG ACCUSED PRODUCTS INFRINGE CLAIM 8 OF THE '915 PATENT**

20 28. In my opinion, each of the Samsung Accused Products meets each and every
21 limitation of claim 8 of the '915 patent literally and, in the alternative, under the doctrine of
22 equivalents, as explained below. Videos of various Accused Products performing the limitations
23 of this claim were included in my Infringement Report as Exhibit 18 (Galaxy Tab 10.1), Exhibit
24 19 (Galaxy S II), Exhibit 20 (Vibrant), and Exhibit 21 (Captivate). They are renumbered as
25 Exhibits 4, 5, 6 and 7 to this Declaration.

26 29. **Claim 8.** Claim 8 recites:

27 A machine readable storage medium storing executable program
28 instructions which when executed cause a data processing system to
perform a method comprising:

1 [a] receiving a user input, the user input is one or more input points
2 applied to a touch-sensitive display that is integrated with the data
processing system;

3 [b] creating an event object in response to the user input;

4 [c] determining whether the event object invokes a scroll or gesture
5 operation by distinguishing between a single input point applied to
6 the touch-sensitive display that is interpreted as the scroll operation
and two or more input points applied to the touch-sensitive display
that are interpreted as the gesture operation

7 [d] issuing at least one scroll or gesture call based on invoking the
8 scroll or gesture operation;

9 [e] responding to at least one scroll call, if issued, by scrolling a
window having a view associated with the event object;

10 [f] responding to at least one gesture call, if issued, by scaling the
11 view associated with the event object based on receiving the two or
more input points in the form of the user input.

12 30. **Claim 8 – Preamble and limitations [a] , [b], [d], [e] and [f]** Each of the
13 Accused Products is either a smartphone or tablet running a version of the Android operating
14 system, which includes a data processing system. Each Accused Product includes a computer
15 readable storage medium storing executable program instructions which when executed cause the
16 data processing system to perform the method described in claim 8. I have previously submitted
17 an Expert Report on the Infringement of the '915 patent, providing details on how the Accused
18 Products meet the preamble (if it is a claim limitation) and every limitation found in Claim 8. In
19 my Infringement Report I discussed method claim 1 in detail, and opined that device claim 8 was
20 infringed for essentially the same reasons as method claim 1, because claim 8 in essence claims a
21 device for performing the method of claim 1. My infringement opinions included claim charts as
22 Exhibit 17 to my Infringement Report, which were submitted as Exhibit 14 to Mr. Gray's
23 Declaration. They also included claim charts as Exhibit 16 to my Infringement Report, which is
24 attached to this Declaration as Exhibit 8. I incorporate those claim charts by reference in this
25 Declaration.

26 31. Because Samsung's Motion and Mr. Gray's Declaration do not contest that the
27 Samsung Accused Products infringe the preamble or the limitations Mr. Gray had labeled as [a],
28 [b], [d], [e] and [f], but instead challenge only whether the Accused Products meet limitation [c], I

1 will focus on that limitation rather than reiterating all of the reasons why the other limitations are
2 present in the Samsung Accused Products.

3 22. **Claim 8 – Element [a] “receiving a user input, the user input is one or more**
4 **input points applied to a touch-sensitive display that is integrated with the data processing**
5 **system.”**

6 23. In my opinion, each of the Accused Products infringes this limitation. The
7 Accused Products receive a user input. The user input includes one or more input points (one or
8 more fingers) applied to the touch-sensitive display that is integrated with the Samsung device.
9 Samsung has not disputed this in its Motion or in Mr. Gray’s Declaration.

10 24. For example, the Galaxy Tab 10.1 and the Galaxy S II receives a user input with
11 one input point (one finger) applied to the touch-sensitive display as illustrated below. The
12 touch-sensitive displays are integrated into the Galaxy Tab 10.1 and the Galaxy S II.

13 25. Based on my observations of the Accused Products, as well as my analysis of the
14 source code for each major release of Android running on the Accused Products (Android 2.1,
15 2.2, 2.3, and 3.1), I have determined that each Accused Product receives a user input, where the
16 user input is one or more input points applied to the touch-sensitive display that is integrated with
17 the device. The claim chart in Exhibit 17 to my Infringement Report identified analogous code
18 that satisfies this element in Android 2.1, 2.2, and 2.3. (Gray Decl. at Ex. 14.)

19 26. **Claim 8 – Element [b] “creating an event object in response to the user**
20 **input.”** In my opinion, each of the Accused Products includes a machine readable storage
21 medium storing executable program instructions which when executed cause a data processing
22 system to create an event object in response to the user input. Samsung has not disputed this in
23 its Motion or in Mr. Gray’s Declaration.

24 27. Each of the Accused Products, via the Android platform on which it operates,
25 creates an event object in response to the user input.

26 28. Under the public Android platform, a MotionEvent object is created in response to
27 a touch on the touch screen. ([http://developer.android.com/reference/android/view/](http://developer.android.com/reference/android/view/MotionEvent.html)
28 [MotionEvent.html.](http://developer.android.com/reference/android/view/MotionEvent.html))

1 39. I have confirmed that code identical or very similar to the public Android code
2 also appears in the Accused Products. For example, in the Galaxy Tab 10.1 tablet, which runs a
3 version of Android 3.1, the user input is processed by the device driver, which passes the input
4 into user space and parses it into an event object referred to as the “MotionEvent” object. This
5 object is an event object created by the method InputConsumer::populateMotionEvent(). (See
6 frameworks/base/libs/ui/inputTransport.cpp:683-712 [SAMNDCA-C000002822]; see also
7 frameworks/base/libs/ui/input.cpp:351-382 [SAMNDCA-C000002830 to -C000002831]
8 (MotionEvent::initialize() method)).

9 40. Based on my observations of the Accused Products, as well as my analysis of the
10 source code for each major release of Android running on the Accused Products (Android 2.1,
11 2.2, 2.3, and 3.1), I have determined that each Accused Product includes similar computer code
12 that creates an event object in response to user input. The claim chart in Exhibit 17 to my
13 Infringement Report identifies analogous code that satisfies this element in Android 2.1, 2.2, and
14 2.3. (Gray Decl. at Ex. 14.)

15 41. **Claim 8 – Element [c] “determining whether the event object invokes a scroll**
16 **or gesture operation by distinguishing between a single input point applied to the touch-**
17 **sensitive display that is interpreted as the scroll operation and two or more input points**
18 **applied to the touch-sensitive display that are interpreted as the gesture operation.”** In my
19 opinion, each of the Accused Products meets this limitation.

20 42. The Accused Products determine whether an event object invokes a scroll or
21 gesture operation by distinguishing between a single input point (one finger) applied to the touch-
22 sensitive display that is interpreted as the scroll operation and two or more input points (more
23 than one finger) applied to the touch-sensitive display that are interpreted as the gesture operation.

24 43. For example, the Galaxy Tab 10.1 tablet distinguishes between a scroll operation
25 when one finger is applied to the touch-sensitive display and a gesture operation when two or
26 more fingers are applied to the touch-sensitive display.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28



(Scroll operation when one input point is applied.)



(Gesture operation when two or more input points are applied.)

44. For example, the Galaxy S II phone distinguishes between a scroll operation when one finger is applied to the touch-sensitive display and a gesture operation when two or more fingers are applied to the touch-sensitive display, as illustrated below:

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28



1. (Scroll operation when one input point is



2. (Gesture operation when two or more input points are applied.)

45. In the Galaxy Tab 10.1 tablet, which runs Android 3.1, the
handleQueuedMotionEvent() method interprets the input points associated with the MotionEvent
object it processes. The handleQueueMotionEvent() method distinguishes between a single input
point (`ev.getPointerCount() == 1`) and two or more input points (`ev.getPointerCount() > 1`). (See
WebView.java:10281-10314 [SAMDNCA-C000002857].) If one input point is detected, the

1 contact is interpreted as a scroll operation in `handleTouchEventCommon()`. (*See*
2 `WebView.java:10312 [SAMNDCA-C000002857].`) If two or more input points are detected, the
3 contact is interpreted as a gesture operation via a call to `handleMultiTouchInWebView()`. (*See*
4 `WebView.java:10302 [SAMNDCA-C000002857]; WebView.java:7887-7944 [SAMNDCA-`
5 `C000002858].`)

6 46. In the Galaxy S II, which runs Android 2.3, the `onTouchEvent()` method interprets
7 the `MotionEvent` object by executing `ev.getPointerCount()` to distinguish between one
8 (`ev.getPointerCount() == 1`) and two or more input points (`ev.getPointerCount() > 1`). (*See*
9 `WebView.java:7576 [SAMNDCA-C000005757-000005758].`) If one input point is detected, the
10 contact is interpreted as a scroll operation and the `onTouchEvent` method continues executing to
11 handle scrolling. (*See* `Webview.java [SAMNDCA-C000005757-000005772.]` If two or more
12 input points are detected, the contact is interpreted as a gesture operation and the
13 `mScaleGestureDetector.onTouchEvent()` is executed to handle scaling. (*See* `WebView.java:7479`
14 `[SAMNDCA-000005758]; [SAMNDCA-000005821-000005824].`)

15 47. Based on my inspection of Samsung source code for each major release of
16 Android running on the Accused Products (Android 2.1, 2.2, 2.3, and 3.1), I have determined that
17 each Accused Product includes similar computer code that distinguishes between a single input
18 point (one finger) applied to the touch-sensitive display that is interpreted as the scroll operation
19 and two or more input points (more than one finger) applied to the touch-sensitive display that are
20 interpreted as the gesture operation. The claim chart in Exhibit 17 to my Infringement Report
21 identifies analogous code that satisfies this element in Android 2.1, 2.2, and 2.3. (Gray Decl. at
22 Ex. 14.)

23 48. I understand that this limitation is the only one that Mr. Gray addresses in his
24 Declaration in support of Samsung’s Summary Judgment Motion. I further understand that Mr.
25 Gray alleges that the `MotionEvent` object does not “invoke” a scroll or gesture operation. I
26 disagree with Mr. Gray, because the Accused Products contain a `MotionEvent` object that
27 “invokes” a scroll or gesture operation, either literally or under the doctrine of equivalents.
28

1 49. I understand that Mr. Gray interprets the term “invoke” to mean that the event
2 object must itself “call” a scroll or gesture operation. In essence, this would mean the event
3 object must itself call the operation with no intervening steps. I disagree with Mr. Gray’s
4 interpretation of “invoke,” as it fails to interpret “invoke” in light of the specification.

5 50. In my opinion, based on the context provided by the ’915 patent specification and
6 claims, a person of ordinary skill in the art would understand that “invoke” means “causes” or
7 “causes a procedure to be carried out.”

8 51. The specification refers multiple times to the “user input invokes a scroll,” which
9 means that user input causes a scroll operation to occur. As one example, the specification
10 discloses, “The method 1000 for providing the scroll hysteresis operation includes transferring a
11 scroll hysteresis call to determine whether a **user input invokes a scroll** at block 1002.” (Arnold
12 Decl. Ex. 85 [’915 Patent] at 10:66-11:2 (emphasis added).) As another example, the
13 specification discloses, “In an embodiment, the library of the framework provides an API for
14 specifying a scroll hysteresis operation to determine whether a **user input invokes** a scroll.” (*Id.*
15 at 22:62-64 (emphasis added.)) Similarly, the specification discloses that the “user input” may
16 “invoke a gesture” on the view associated with the user input. (*Id.* at 13:37-39.) One skilled in
17 the art would understand that “user input” cannot itself “call” scroll or gesture operation code, but
18 instead causes the scroll or gesture operation to occur via intervening hardware detection and
19 software steps. Mr. Gray’s narrow construction of “invokes” is contrary to the specification.

20 52. Similarly, the specification discloses that the “software invokes an animation that
21 performs a scaling transform on the view associated with the user input.” (Arnold Decl. Ex. 85
22 [’915 Patent] at 15:4-6.) One skilled in the art would understand. In light of this description, that
23 the software event object does not or need not call a scroll or scaling function directly, but rather
24 that the general code for scrolling and scaling would perform those operations.

25 53. I disagree with Mr. Gray’s assertion that there is a significant distinction in the
26 ’915 patent between “event object invokes” and “user input invokes.” (*See* Gray Decl. ¶ 25.)
27 The patent specification often refers to “event object” as shorthand for “user input in an event
28 object.” As explained in the specification, “A multi-touch driver of the device receives the user

1 input and packages the event into an event object.” (*See* Arnold Decl. [’915 Patent] at 12:30-32.)
2 The previous limitation of claim 8, “creating an event object in response to the user input,”
3 indicates that the event object includes the user input information. Moreover, the interchangeable
4 use of “event object” and “user input” is illustrated throughout the specification, which refers to
5 the “event object,” “user input,” and “software” to invoke, or cause, various operations. (*Id.* at
6 10:16-11:12, 13:37-39, 15:4-6, 22:62-64.)

7 54. The specification also expressly describes how the event object is used to cause
8 scroll or gesture operations to execute via multiple steps. In each of the below examples, the
9 event object does not itself call scroll or gesture operation code, but the event object’s user input
10 information is used to cause the scroll or gesture operation code to execute. For example, the
11 specification discloses, “A window server receives the **event object** and determines whether the
12 event object is a gesture event object. If the window server determines that a gesture event object
13 has been received, then **user interface software issues or transfers the handle gesture call** at
14 block 1302 to a software application associated with a view.” (Arnold Decl. Ex. 85 [’915 Patent]
15 at 12:32-37 (emphasis added).) In this way, the user input information in the event object causes
16 the gesture operation to execute. Similarly, the specification discloses, “If the **events are hand**
17 **events based on a user input, the events are routed to the window** they occurred over. The
18 **window then routes these events to the appropriate control by calling the instance's mouse**
19 **and gesture methods**. The control that receives a mouse down or mouse entered function will
20 continue to get all future calls until the hand is lifted. **If a second finger is detected, the gesture**
21 **methods or functions are invoked.**” (*Id.* at 12:9-16 (emphasis added).) In this way, the event
22 object is based on the user input, causes a routing from the window to the user interface control,
23 which then calls a scroll or gesture operation based upon whether the event object contains a one-
24 finger or two-finger user input. The specification is entirely consistent with my interpretation of
25 “invokes.” Mr. Gray’s narrow definition limiting “invokes” to the event object itself calling a
26 scroll or gesture operation with no intervening steps contradicts the specification.

27 55. My opinion that “invokes” means “causes” or “causes a procedure to be carried
28 out” is supported by dictionaries published near the time of the ’915 patent invention. For

1 example, the Oxford English Dictionary defines “invoke” in the “Computing” context as “cause
2 (a procedure) to be carried out.” (Oxford Dictionary of English (2d Ed.) (2005) attached as
3 Exhibit 9.) As another example, Merriam Webster’s Dictionary defines “invoke” as “to bring
4 about, cause.” (Merriam Webster’s Dictionary (2004) attached as Exhibit 10.)

5 56. I disagree with Mr. Gray’s statement that MotionEvent object never invokes a
6 scroll or gesture operation. (*See* Gray Decl. ¶ 35.) Mr. Gray does not dispute my conclusion that
7 data indicating “one touch” or “more than one touch” in an “event object” causes Samsung’s code
8 to take different paths: “one-touch” for scrolling and “multi-touch” for scaling. (Bartlett Decl.
9 Ex. 30 at 38:4-50:6.)

10 57. Mr. Gray attempts to shift focus onto the WebView class, which is merely an
11 abstraction for a set of code that assists with routing event information to the window animation
12 software that executes the scrolling or scaling. (*See* Gray Decl. ¶ 34-35.) Mr. Gray never
13 contests that Samsung code calls the scroll or gesture operation depending upon the number of
14 touch points, but instead alleges it is WebView, not the event object, that calls the scroll or
15 gesture operation. I disagree with Mr. Gray. The specification clearly discloses that the window
16 or user interface code would use the event object’s user input information to cause a call to the
17 scroll or gesture operation. As I explained above, the specification discloses that the window
18 class (*i.e.*, WebView) calls the scroll and gesture methods using the event: “If the events are hand
19 events based on a user input, the **events are routed to the window** they occurred over. The
20 **window then routes these events to the appropriate control by calling the instance's mouse**
21 **and gesture methods.**” (Arnold Decl. [’915 Patent] at 12:9-16 (emphasis added).) Further, the
22 specification explains that the window’s user interface software (*i.e.*, WebView) issues or
23 transfers the gesture call based on the event: “A window server receives the event object and
24 determines whether the event object is a gesture event object. If the **window server** determines
25 that a gesture event object has been received, then **user interface software issues or transfers**
26 **the handle gesture call** at block 1302 to a software application associated with a view.” (*Id.* at
27 12:32-37 (emphasis added).) Thus, the ’915 patent specification clearly shows that “invokes”
28 includes the use of additional code (such as WebView) to assist with routing the event object

1 information and, depending on whether the event object indicates there is one or more input
2 points, call the scroll or gesture operation.

3 58. As I explained above, the MotionEvent code (ev.getPointerCount()) executes, and,
4 depending on whether its code indicates there is one or more input points, causes either the scroll
5 operation code (handleTouchEventCommon()) or gesture operation code
6 (handleMultiTouchInWebView()) to execute. (See, e.g., WebView.java:10281-10314
7 [SAMDNCA-C000002857]; WebView.java:7887-7944 [SAMNDCA-C000002858].) This is
8 precisely what it means to “determin[e] whether the event object invokes a scroll or gesture
9 operation by distinguishing a single input point . . . and two or more input points . . . ,” as recited
10 in claim 8[c].

11 59. I also disagree with Mr. Gray’s comment that the MotionEvent object is a passive
12 container of information. (See Gray Decl. ¶ 33.) The MotionEvent object’s getPointerCount()
13 method is executed to identify the number of user inputs as part of the determination of whether
14 to call a scroll or gesture operation. (See, e.g., WebView.java:10281-10314 [SAMDNCA-
15 C000002857].)

16 60. I further understand that Samsung alleges that the “named inventors agree with its
17 construction,” but neither co-inventor agreed that “invoke” was limited to the event object itself
18 calling a function. Co-inventor Mr. Herz testified, “[A]n example of invoking something would
19 be . . . *causing that code to run.*” (Bartlett Decl. Ex. 53 at 95:15-19.) Herz also testified, “*So in*
20 *the example I gave invoking could mean, yeah, causing something to happen.*” (*Id.* at 95:20-
21 96:1.) Further, in response to a question, “Is there a way the view can determine whether the
22 event is associated with a scroll or gesture operation,” Mr. Herz testified “There is code that
23 looked at a stream of events and determines whether or not it should scroll or do other things. . . .
24 [T]hat code is called as a result of the events going to the view.” (*Id.* at 76:19-77:8.)
25 Additionally, Mr. Herz explained: “So the code I would have written would have taken event
26 input, and it would have looked at that and tried to decide whether or not we should scroll a – the
27 contents of a field.” (*Id.* at 147:4-11.) Contrary to Mr. Gray’s out-of-context citation, Mr. Herz
28

1 supports my interpretation that the “event object” does not itself call the scroll or gesture
2 operation code, but instead causes the scroll or gesture operation to occur.

3 61. In addition, the other co-inventor, Mr. Platzer qualified his statement, “With
4 regards to the patent, I’m not comfortable in defining ‘invoked.’ But as an example, within
5 UIKit, there is code that does look at the number of points and does decide on a gesture or scroll.
6 . . . There is code within UIKit that looks at the number of touch points and decides which code to
7 call it based on the number of touch points.” (Bartlett Decl. 54 at 84:13-22.) Mr. Platzer never
8 testified that the event object needed to itself call a scroll or gesture operation.

9 62. Samsung’s Motion mischaracterizes my extensive testimony on the meaning of
10 “invoke,” and even deletes a key sentence for the snippet of testimony it does quote. I explained
11 multiple times that the meaning of “invokes” depends on the context. (Gray Decl. Ex. 7 at
12 313:14-319:15.)

13 63. Further, I note that Samsung and Mr. Gray have been unclear and inconsistent as
14 to the meaning of “invoke” and how it might relate to an “event object.” Samsung initially
15 asserted, after fact discovery closed, that Apple had failed to prove “determining whether the
16 event object invokes a scroll or gesture operation, as ‘event objects’ are incapable of invoking
17 operations.” This assertion would be mean that “invoke” should be interpreted as an
18 impossibility, which contradicts the ’915 patent specification.

19 64. Moreover, in his expert report, Mr. Gray states, “In my 35 years of systems
20 experience, I have never observed a system where an event object invoked a method.” (Bartlett
21 Decl. Ex. 31 at ¶ 266.) As I observed in my Validity Report, the construction that an event
22 object *never* invokes a method appears to be inconsistent with the notion that it would have been
23 obvious for a person of ordinary skill to add an event object that calls a scroll operation. (Ex. 2
24 at ¶ 177, 202, 226.) At his deposition, Mr. Gray retracted his earlier position, identifying his
25 paragraph 266 as an error in his Invalidity Report: “That’s not true. That’s the inaccuracy.”
26 (Bartlett Decl. Ex. 30 at 52:20-53:11.) I agree that Mr. Gray’s earlier position in his Invalidity
27 Report was inaccurate, and I think his current position is inaccurate as well.

1 65. I further note that Mr. Gray also retracted his earlier reference to Mr. Platzer's
2 testimony. Mr. Gray earlier said that Platzer "agreed" with him in his expert report but changed
3 his opinion at the deposition. (Bartlett Decl. Ex. 30 at 53:12-19.) I agree that Mr. Gray's earlier
4 statement was inaccurate, and I think his new position again saying that Platzer "supports" his
5 opinion also is incorrect. (*See id.* at 53:12-19; Gray Decl. ¶ 22.)

6 66. For the above reasons, it is my opinion that the Accused Products literally infringe
7 claim 8[c] as each is a machine readable medium containing instructions that "determine[e]
8 whether the event object invokes a scroll or gesture operation by distinguishing between a single
9 input point applied to the touch-sensitive display that is interpreted as the scroll operation and two
10 or more input points applied to the touch-sensitive display that are interpreted as the gesture
11 operation."

12 67. To the extent that this limitation is not met literally, in my opinion it is met under
13 the doctrine of equivalents. I disagree with Samsung's proposed claim construction for "invoke,"
14 but even if I were to adopt Samsung's new proposed construction, the Accused Products infringe
15 under the doctrine of equivalents because each of the Accused Products is a machine readable
16 medium containing instructions that perform steps insubstantially different from "determining
17 whether the event object invokes a scroll or gesture operation by distinguishing between a single
18 input point applied to the touch-sensitive display that is interpreted as the scroll operation and two
19 or more input points applied to the touch-sensitive display that are interpreted as the gesture
20 operation."

21 68. It also is my opinion that, even if I were to adopt Samsung's new proposed
22 construction, the Accused Products infringe under the doctrine of equivalents because they
23 perform substantially the same function in substantially the same way to obtain substantially the
24 same result.

25 69. First, it is my opinion that the Accused Products perform substantially the same
26 function as the recited limitation. The function of the limitation is "*determining* whether the
27 event object invokes a gesture operation by distinguishing between a single input point applied to
28 the touch-sensitive surface display that is interpreted as the scroll operation and two or more input

1 points applied to the touch-sensitive display that are interpreted as the gesture operation.” (’915
2 patent claim 8[c] (emphasis added).) In the context of the ’915 patent specification and claim 8,
3 the function is *determining* based on distinguishing between one or two or more user input points
4 in the event object whether a scroll or gesture operation should execute. The functions are the
5 same.

6 70. Second, the Accused Products perform this function in substantially the same way
7 as in the claim limitation. The Accused Products all perform a logical test using the event
8 object’s user input information. For example, in Android 3.1, the MotionEvent code (ev refers to
9 MotionEvent) executes to distinguish between a single input point (ev.getPointerCount() == 1)
10 and two or more input points (ev.getPointerCount() > 1). (*See, e.g.,* WebView.java:10281-10314
11 [SAMDNCA-C000002857].) In addition, the logical pathway occurs in the same order: the event
12 object code executes first, followed by either the scroll operation code or gesture operation code.
13 (*See, e.g.,* SAMNDCA-C000002857-000002858.)

14 71. I disagree with Mr. Gray’s characterization that my position eliminates the “event
15 object” limitation. (*See* Gray Decl. ¶ 44.) The MotionEvent object’s getPointerCount() method
16 is executed as part of the determination of whether to call a scroll or gesture operation. (*See, e.g.,*
17 WebView.java:10281-10314 [SAMDNCA-C000002857].) Moreover, the user input data is
18 contained within the event object. Thus, the “event object” is used as part of the “way” to
19 achieve this limitation.

20 72. Finally, the Accused Products obtain substantially the same result, *i.e.*, the
21 execution of either the scroll operation or gesture operation code, depending on whether there is a
22 single input point or two or more input points.

23 73. For the above reasons, it is my opinion that the Accused Products infringe claim
24 8[c] under the doctrine of equivalents as each is a machine readable medium containing
25 instructions that perform the equivalent of “determining whether the event object invokes a scroll
26 or gesture operation by distinguishing between a single input point applied to the touch-sensitive
27 display that is interpreted as the scroll operation and two or more input points applied to the
28 touch-sensitive display that are interpreted as the gesture operation.”

1 74. **Claim 8 – Element [d] “issuing at least one scroll or gesture call based on**
2 **invoking the scroll or gesture operation.”** In my opinion, each of the Accused Products meets
3 this limitation. I understand that Samsung and Mr. Gray have not disputed this point.

4 75. The images reproduced above show the Galaxy 10.1 tablet and the Galaxy S II
5 smartphone issuing a scroll call when the scroll operation is invoked and issuing a gesture call
6 when the gesture operation is invoked. The software steps are summarized below.

7 76. For example, in the Galaxy 10.1 tablet, if one input point is detected,
8 `handleQueuedMotionEvent()` will call `handleTouchEventCommon()` (`WebView.java:10312`
9 [`SAMNDCA-C000002926`]), which issues a scroll call to `doDrag()` or `doFling()`.
10 (`WebView.java:7617, 7772` [`SAMNDCA-C000002926, -C000002930`]) If two or more input
11 points are detected, the contact is interpreted as a gesture operation and a call to
12 `handleMultiTouchInWebView()` is made. (*See* `WebView.java:10302` [`SAMNDCA-`
13 `C000002857`]; `WebView.java:7887-7944` [`SAMNDCA-C000002858`].)

14 77. Based on my inspection of Samsung source code for each major release of
15 Android running on the Accused Products (Android 2.1, 2.2, 2.3, and 3.1), I have determined that
16 each Accused Product includes similar computer code that issues at least one scroll or gesture call
17 based on invoking the scroll or gesture operation. The claim chart in Exhibit 17 to my
18 Infringement Report identifies analogous code that satisfies this element in Android 2.1, 2.2, and
19 2.3. (Gray Decl. at Ex. 14.)

20 78. **Claim 8 – Element [e] “responding to at least one scroll call, if issued, by**
21 **scrolling a window having a view associated with the event object.”** In my opinion, each of
22 the Accused Products meets this limitation, and I understand that Samsung and Mr. Gray have not
23 contested this point. The images of the Galaxy 10.1 tablet and Galaxy S II phone performing
24 scrolling reproduced above demonstrate the performance of this limitation. The software steps
25 are discussed below.

26 79. For example, in the Galaxy 10.1 tablet, the `handleTouchEventCommon()` method
27 calls `doFling()` for a scroll operation. (*See* `WebView.java:7272-7821` [`SAMNDCA-C000002919`
28 to `-C000002931`] (call done at 7772).) `doFling()` then calls the `Overscroller.fling()` method. (*See*

1 WebView.java:9236-9376 [SAMNDCA-C000002932 to -C000002935].) OverScroller.fling()
2 itself calls two instances of the SplineOverScroller class, each of which is responsible for
3 scrolling in one axis (i.e., one scrolls horizontally and the other scrolls vertically). (See
4 OverScroller.java:406-448 [SAMNDCA-C000002945].) The SplineOverScroller class thus
5 maintains state information for the fling. (See *id.*)

6 80. The SplineOverScroller class tracks the start points, start time, duration, total
7 distance, and the final position for the fling. (OverScroller.java:748-782 [SAMNDCA-
8 C000002952 to -C000002953].) The SplineOverScroller.fling() function determines the final
9 position of the fling before beginning the fling operation begins.

10 81. The actual rendering of the fling occurs subsequently as part of the drawing cycle.
11 At the end of an event processing cycle, the method computeScroll() is called to compute which
12 part of the view should be rendered to the user. (See WebView.java:3568-3654 [SAMNDCA-
13 C000002958 to -C000002959]. The computeScroll() method uses the SplineOverScroller class
14 to extract the state information for the fling. (See *id.*) Afterwards, it calls
15 WebView.overScrollBy() to scroll the content. (See *id.*; see also View.java:11663-11715
16 [SAMNDCA-C000002960 to -C000002961] (WebView.overScrollBy()).) onOverScrollBy()
17 itself calls onOverScroller() to ensure the intended scroll coordinates are valid and then calls
18 View.scrollTo(). (See View.java:11663-11715 [SAMNDCA-C000002960 to -C000002961];
19 WebView.java:3130-3162 [SAMDNCA-2962].) View.scrollTo() scrolls the window by setting
20 mScrollX and mScrollY. (See WebView.java:3130-3162 [SAMDNCA-2962].)

21 82. Alternatively, it is my opinion that the scrolling occurs when the
22 WebView.onDraw() method is subsequently called to translate and draw the view shown to the
23 user. (See WebView.java:4261-4418 [SAMNDCA-C000002965 to -C000002968] (with call to
24 trackFPS() at 4416); WebView.java:8757-8791 [SAMNDCA-C000002964] (trackFPS()
25 translates based on mScrollX and mScrollY then draws).)

26 83. Based on my inspection of Samsung source code for each major release of
27 Android running on the Accused Products (Android 2.1, 2.2, 2.3, and 3.1), I have determined that
28 each Accused Product includes similar computer code that responds to at least one scroll call by

1 scrolling a window having a view associated with the MotionEvent. The claim chart in Exhibit
2 17 to my Infringement Report identifies analogous code that satisfies this element in Android 2.1,
3 2.2, and 2.3. (Gray Decl. at Ex. 14.)

4 84. **Claim 8 – Element [f] “responding to at least one gesture call, if issued, by**
5 **scaling the view associated with the event object based on receiving the two or more input**
6 **points in the form of the user input.”** In my opinion, each of the Accused Products meets this
7 limitation, which Samsung and Mr. Gray do not dispute.

8 85. For example, the Galaxy 10.1 tablet and the Galaxy S II phone will respond to at
9 least one gesture call by scaling the view (zooming) associated with the MotionEvent object
10 based on receiving two or more input points in the form of the user input, as shown in the
11 “scaling” images reproduced above. The software steps are discussed below.

12 86. For example, in the Galaxy 10.1 tablet, the handleMultiTouchInWebView()
13 method calls the WebViewScaleGestureDetector.onTouchEvent() method to perform the scaling
14 (zoom) operation using the MotionEvent object information, which includes the two or more
15 input points touching the screen. (See WebViewScaleGestureDetector.java:189 [SAMNDCA-
16 C000002905].) onTouchEvent() calls setContext(), which records information about the position
17 of the two input points corresponding, for example, to the user’s fingers on the screen
18 (WebViewScaleGestureDetector.java:581-630 [SAMNDCA-C000002524 to -C000002525]). As
19 the user moves his fingers relative to one another—as in, for example, a pinching or de-pinching
20 gesture—the handleScale() method of the ZoomManager class calls the
21 WebViewScaleGestureDetector’s getScaleFactor() method to calculate the scale factor based on
22 the ratio of the current distance between the fingers and the previous distance between them (as of
23 the last time the touch screen was polled for input). (ZoomManager.java:1323 [SAMNDCA-
24 C000002410]; WebViewScaleGestureDetector.java:763-768 [SAMNDCA-C000002528].)
25 handleScale() then calls setZoomScale(), which uses the calculated scale factor to scale the
26 WebView and all of its child views. ZoomManager.java:1372 [SAMNDCA-C000002411];
27 ZoomManager.java:851-949 [SAMNDCA-C000002399 to -C000002402].)

1 87. Based on my inspection of Samsung source code for each major release of
2 Android running on the Accused Products (Android 2.1, 2.2, 2.3, and 3.1), I have determined that
3 each Accused Product includes similar computer code that responds to at least one gesture call, if
4 issued, by scaling the view associated with the event object based on receiving the two or more
5 input points in the form of the user input. The claim chart in Exhibit 17 to my Infringement
6 Report identifies analogous code that satisfies this element in Android 2.1, 2.2, and 2.3. (Gray
7 Decl. Ex. 14.)

8
9 **VI. LAUNCHTILE DOES NOT INVALIDATE CLAIM 50 OF THE '163 PATENT**

10 88. Claim 50 of the '163 patent recites¹:

11 A portable electronic device, comprising:

12 [a] a touch screen display; one or more processors; memory; and one or more
13 programs, wherein the one or more programs are stored in the memory and
 configured to be executed by the one or more processors,

14 [b] the one or more programs including: instructions for displaying at least
15 a portion of a structured electronic document on the touch screen display,
 wherein the structured electronic document comprises a plurality of boxes
 of content;

16 [c] instructions for detecting a first gesture at a location on the displayed
17 portion of the structured electronic document; instructions for determining
18 a first box in the plurality of boxes at the location of the first gesture;
 instructions for enlarging and translating the structured electronic
 document so that the first box is substantially centered on the touch screen
19 display;

20 [d] instruction[s] for, while the first box is enlarged, a second gesture is
21 detected on a second box other than the first box; and instructions for, in
22 response to detecting the second gesture, the structured electronic
 document is translated so that the second box is substantially centered on
 the touch screen display.

23 89. As stated in my Validity Report (attached as Exhibit 2) at ¶¶ 29-38, it is my
24 opinion that Claim 50 of the '163 patent is not anticipated or otherwise invalidated by the
25 LaunchTile System, the LaunchTile Publication describing it (Bederson Decl., Ex. A), or the
26 XNav System (which operates substantially identically to the LaunchTile System for purposes
27

28 ¹ I adopt the separation of claim elements set forth in Mr. Gray's Declaration.

1 relevant to this litigation). For convenience, I will refer collectively to these three pieces of
2 alleged prior art as “LaunchTile” except when discussing content or functionality that is specific
3 to one of them.

4 90. I incorporate here by reference the arguments for validity over LaunchTile made in
5 paragraphs 29-38 of my Validity Report. (Ex. 2.) I made these arguments in my Validity Report
6 in the context of claim 2, but they apply equally to claim 50, which has claim limitations
7 substantially identical to those in claim 2. Mr. Gray agrees that “Claim 2 is a ‘computer
8 implemented method’ claim, and generally tracks the language of independent claims 49, 50, 51,
9 and 52.” (Bartlett Decl. Ex. 31 ¶ 289.) Claim 50 requires one or more programs including
10 “instructions for” performing each of the method steps described in claim 2. LaunchTile fails to
11 anticipate claim 50 for at least the same reasons that it fails to anticipate claim 2.

12 **A. Overview of LaunchTile**

13 91. LaunchTile is a research prototype system that provides the ability to launch 36
14 applications via tiles presented using a display abstraction that its authors call an “interactive
15 zoomspace.” (Bederson Decl., Ex. A at 204.) The zoomspace provides three levels of display: the
16 World View, which displays application tiles (symbolic visual representations) corresponding to
17 each of the 36 applications in a 6-by-6 grid; the Zone View, which displays four application tiles
18 with additional application-related content in a 2-by-2 grid; and the Application View, which
19 launches and allows a user to interact with each application itself. Clicking or tapping on a
20 location in the World View initiates an animation that fills the screen with a Zone View
21 rendering—distinct from the content displayed in the corresponding portion of the World View—
22 of the four application tiles around the location of the user’s touch. Once the Zone View is
23 rendered, clicking or tapping on one of the four displayed application tiles launches the
24 application—for example, an email client application or a mapping application—to which the
25 selected application tile corresponds.

26 92. As the above description suggests, LaunchTile targets an entirely different
27 problem from the one that is solved by claim 50 of the ’163 patent: LaunchTile addresses the use
28 of a fixed set of applications in a predefined layout, whereas the ’163 patent deals with reading

1 and navigating arbitrarily-sized structured electronic documents on a small screen. LaunchTile
2 creates substantively different renderings of a fixed set of iconic application tiles. These tiles
3 facilitate the launching of the applications to which the tiles correspond. Claim 50, by contrast,
4 applies enlargement and translation to a unified, but arbitrarily sized, structured electronic
5 document to aid a user’s viewing of areas of interest in that document. LaunchTile fails to
6 disclose multiple elements of claim 50, as the analysis that follows will demonstrate.

7 **B. Claim 50, Element [b]**

8 93. LaunchTile does not disclose the element of claim 50 of “instructions for
9 displaying at least a portion of a structured electronic document on the touch screen display,
10 wherein the structured electronic document comprises a plurality of boxes of content.”

11 94. According to Mr. Gray, LaunchTile displays a structured electronic document in
12 the World View. (Gray Decl. ¶ 76 (“It is my opinion that this 6x6 zoomspace is a ‘structured
13 electronic document’ with 36 embedded Application tiles, each of which is also a structured
14 electronic document.”).) I disagree. As I opined in my deposition in response to Samsung’s
15 questioning on this point (Gray Decl. Ex. 6 at 171:14-176:15), the mere fact that LaunchTile
16 arranges a set of otherwise conceptually independent application tiles into a grid for display does
17 not automatically qualify that collection as a single electronic document.

18 95. In my opinion, those of skill in the art of the ’163 patent would understand that
19 there must be some conceptual relationship or commonality in the information in a “document”
20 that is sufficient to justify treating that information as a single, discrete entity. For electronic
21 documents, the classic indication of such a relationship is the storage of information in a single
22 file, such as a text file, an image file, an HTML file, or a spreadsheet file. Where a single file is
23 not present, information must be related in a conceptually equivalent way to be considered a
24 “document.”

25 96. As I discussed in my Infringement Report, the display of a web page, such as the
26 *New York Times* home page, in a mobile device’s web browser is a paradigm example of the
27 display and navigation of a structured electronic document that the ’163 patent targets. The *New*
28 *York Times* home page is a structured electronic document that includes several boxes of content

1 that mobile devices—including Apple’s iOS products and the Samsung Accused Products—can
2 display on their touch screen displays. These devices detect a user’s double tap gesture (two taps
3 on the touch screen in quick succession) on a box of content, and respond to that gesture by
4 determining which box was tapped and then enlarging and translating the web page to
5 substantially center that box on the screen. If the user proceeds to double tap on a second box of
6 content on the web page, the web page is translated to substantially center that second box on the
7 screen.

8 97. The various application tiles that LaunchTile is programmed to display together to
9 create the World View screen are not one electronic document. The different applications that
10 these tiles represent are entirely conceptually independent of one another: they are separate
11 programs designed to run independently and accomplish different tasks. LaunchTile purposely
12 uses different levels of abstraction to provide three different layers of information about a fixed
13 number of application programs. At each layer the system displays different content distinct from
14 the content in other layers, and it launches distinct application programs when an individual tile is
15 touched.

16 98. My review of the XNav source code, which I understand is functionally equivalent
17 in the relevant respects to the code for the LaunchTile, confirms that the “interactive zoomspace”
18 that Mr. Gray identifies as the structured electronic document displayed in the World View is
19 actually just a programmatically assembled collection of separate image files representative of the
20 36 disparate applications displayed in the World View. This is consistent with Dr. Bederson’s
21 own description of LaunchTile. (Bederson Decl. ¶ 14 (“In our prototype implementation, the
22 individual tiles in LaunchTile were typically represented by one or more image files (.png
23 files).”))

24 99. Dr. Bederson and Mr. Gray attempt to manufacture a connection between the
25 various application tiles in the World View by resorting to the idea that LaunchTile’s code for
26 displaying these separate pieces assembles all of them into a “single, hierarchical object oriented
27 data structure.” (Bederson Decl. ¶ 13; cited by Gray Decl. ¶ 77.) Such a data structure—which is
28 a programming construct *created by LaunchTile* to facilitate display that could be populated with

1 arbitrary, unrelated content—is not an “electronic document” within the meaning of the ’163
2 patent because it lacks the prior semantic association of its contents that a “document” requires.

3 100. Dr. Bederson’s inapt comparison of LaunchTile’s display-facilitating “data
4 structure” to an HTML document (Bederson Decl. ¶ 13:9-12) highlights a key distinction between
5 LaunchTile’s operation and the display of a true electronic document, such as the rendering of an
6 HTML document in a web browser. Dr. Bederson claims that LaunchTile’s “**creating** [a] single,
7 hierarchical object oriented data structure that is then translated into the visual representation
8 displayed to the user” (emphasis added) is “similar to the process that occurs when a typical web
9 browser application interprets and transforms the elements of a standard HTML document into
10 what is known as a data object model that can then be visually presented to the user as a single,
11 unified web page.” (*Id.*) A web browser, however, does not “creat[e]” the HTML documents
12 that it displays. Rather, unlike LaunchTile—which creates at runtime (i.e., when the program is
13 executed and the display is rendered) the “single object-oriented data structure” to which Dr.
14 Bederson and Mr. Gray refer²—a web browser takes as input a discrete quantum of information
15 that is already semantically associated as a unified HTML “document.” That a browser performs
16 additional processing to render an HTML document into viewable form does not change the fact
17 that it takes a unified HTML document as input, while LaunchTile merely assembles, for display
18 purposes, disparate image resources representative of independent applications.

19 101. For the reasons above, it is my opinion that the World View in LaunchTile does
20 not “display[] at least a portion of a structured electronic document” and therefore does not
21 disclose this element of claim 50.

22 **C. Claim 50, Element [c]**

23 102. LaunchTile does not disclose the element of claim 50 of “instructions for detecting
24 a first gesture at a location on the displayed portion of the structured electronic document;
25 instructions for determining a first box in the plurality of boxes at the location of the first gesture;

26
27 ² Dr. Bederson is careful to limit his testimony to say only that “embedded tiles were
28 always part of one unified zoomspace that was dependent on a single object-oriented data
structure for its content **during the rendering process.**” (Bederson Decl. ¶ 14 (emphasis added.))
No such unifying structure exists prior to—or independently of—LaunchTile’s being executed.

1 [and] instructions for enlarging and translating the structured electronic document so that the first
2 box is substantially centered on the touch screen display.”

3 103. Mr. Gray opines that “LaunchTile’s instructions for displaying an animated
4 transition from World view to Zone view” disclose this element of claim 50. (Gray Decl. ¶ 85.)
5 As I discussed in my Validity Report, I disagree. Claim 50 requires that the “structured electronic
6 document” that is “enlarge[ed] and translate[ed]” (such that the enlarged portion of it is
7 “substantially centered on the touch screen display”) must be the same structured electronic
8 document that includes a location where “a first gesture” is detected and “a first box” is
9 determined. LaunchTile does not meet this requirement because the content displayed in the
10 World View is entirely replaced with different content, assembled by LaunchTile from different
11 underlying graphical assets, when the system transitions from World View to Zone View. Total
12 replacement of content does not meet any reasonable definition of “enlarging and translating,”
13 and any “box” that existed in the World View is not part of the Zone View. The images in
14 Exhibit 3 to Mr. Gray’s Declaration plainly show entirely different content displayed before and
15 after the transition from World View to Zone View. For example, the single phone icon in the
16 World View becomes a list of calls in the Zone View; the email and calendar cells similarly
17 become detailed lists in the Zone View where they were merely iconic representations in the
18 World View:



25 104. Mr. Gray attempts to unify the entirely different content of the World View and
26 Zone View into a single structured electronic document by claiming that the “zoomspace” itself is
27 the structured electronic document of interest. (Gray Decl. ¶ 85.) But the zoomspace is not an
28 “electronic document” at all. It is, rather, an abstraction that refers to different possible

1 renderings of a set of independent application tiles. As discussed in the previous section in the
2 World View context, it is the LaunchTile program itself that assembles, at runtime, separate
3 image files representative of independent applications into the different grids that are displayed in
4 the World and Zone Views. None of these layouts are dictated by any discrete “document” in the
5 way that, for example, a preexisting HTML file encapsulates a web page that is rendered in a web
6 browser.

7 105. Mr. Gray mischaracterizes my deposition testimony stating that an electronic
8 document is “usually some cohesive piece of information” to claim that it supports his conclusion
9 that “the embedded tiles at the World view are part of the same ‘document’ rendered in further
10 detail at Zone view.” (Gray Decl. ¶ 88.) My statement cuts exactly the opposite way, supporting
11 the contrary conclusion that there is no “document” in common across the World and Zone
12 Views. The separate image resources representative of independent applications that LaunchTile
13 assembles do not together constitute “some cohesive piece of information” on their own. The
14 hierarchical relationship between them, on which Mr. Gray depends, is only imposed by
15 LaunchTile itself as the program executes. This indicates the lack of any true “document” on
16 display across the different levels of LaunchTile, not the presence of one.

17 106. Even if one considered the programmatically imposed layout of four application
18 tile images in a given Zone View a structured electronic document, it would still not be the same
19 structured electronic document as anything displayed in the World View. As the images above of
20 the World-to-Zone View transition show and my review of XNav code confirms, LaunchTile
21 assembles each 2-by-2 tile Zone View from image resources different from those that represent
22 the same applications in the World View. The mere conceptual association with the same
23 underlying applications is not enough, in my opinion, to qualify these otherwise entirely distinct
24 renderings of content as the same structured electronic document. Because the content of any
25 given set of four application tiles displayed in the World View is completely different from the
26 tiles representing those same applications in the Zone View, it is clear that the Zone View
27 rendering is not the result of simply “enlarging and translating” any purported structured
28 electronic document displayed in the World View.

1 107. In his deposition, Mr. Gray admitted that the “Zone View” does not result from
2 enlarging and translating the same “structured electronic document” from which the box of
3 content was selected:

4 Q. Is it your opinion that the transition from the world view to the
5 zone view shown on page 4 of your Appendix 7 [the LaunchTile
6 invalidity claim chart], that what has been enlarged and translated is
7 the same structured electronic document that is shown in the world
8 view?

9 A. **No.** The document which is being shown in the zone view on
10 page 4 is a box of content from the structured electronic document
11 shown in the world view.

12 (Bartlett Decl. Ex. 30 at 205:21-206:3 (emphasis supplied).)

13 In responding to a follow-up question, Mr. Gray confirmed that the Zone View was not an
14 “enlarging” of the four tiles as they were displayed in the World View:

15 Q. It’s different content. It’s not simply an enlarging of the images
16 that are shown in the tile in the world view; it is a –looking at
17 different data and displaying different data rather than displaying
18 the same thing in a larger font size or a larger image, right?

19 [Objection; argumentative, misstates the document]

20 A. Let me agree that it is not a magnification of what’s in the—in
21 the upper right hand corner of the first box of the world view. It is
22 not a magnification—the upper right hand corner of the zone view
23 is not a magnification of the original. That’s accurate.

24 (*Id.* at 209:12-25.)

25 108. Dr. Bederson invokes a concept he calls “semantic zooming” to try to justify his
26 conclusion that “[t]he four tiles that happen to be displayed in Zone view are the same embedded
27 Application tiles (albeit rendered in further detail) that were present at World view.” (Bederson
28 Decl. ¶¶ 17-18.) Dr. Bederson describes a semantic zooming object as “a procedural object that
renders itself differently depending on its viewing size,” and he contends that LaunchTile uses
such objects. (*Id.* ¶ 18.) In my opinion, a semantic zooming object that renders itself entirely
differently across two level of zoom does not meet claim 50’s requirement of “enlarging and
translating” a single structured electronic document to effect the transition between the two
levels.

1 109. Mr. Gray states that “even if the tiles (embedded structured electronic documents)
2 were entirely replaced during the enlarging and translating step, this would not change [his]
3 opinion that the zoomspace, within which the tiles exist at any particular level of zoom, is the
4 same structured electronic document throughout the navigation process.” (Gray Decl. ¶ 89.) He
5 goes on to state:

6 In a somewhat analogous situation, I am aware that web pages
7 (structured electronic documents encoded in HTML) sometimes
8 contain embedded objects displaying live content in the form of
9 advertising material, stock quotes, or "breaking news" headlines.
10 When a user manually refreshes the web page or, in some cases,
11 after some pre-set amount of time, the embedded object will be
updated or even replaced with entirely new different content.
However, no person of ordinary skill in the art would believe that
they were viewing a different webpage (i.e., "structured electronic
document") merely because the content in one embedded element
had changed. (*Id.*)

12 110. The “live content” that Mr. Gray identifies that “refreshes” within a web page is
13 not analogous to the replacement of content that occurs in LaunchTile on the transition from
14 World View to Zone View. In the web-page-live-content scenario, the same HTML document is
15 displayed before and after the refresh of embedded content. As explained above, LaunchTile has
16 no cross-transition analogue to the HTML document, because each level of its display is
17 assembled programmatically from disparate entities and is merely a fixed layout independent of
18 the inherent structure of a single document. In the web page case, the parts of the displayed
19 HTML document other than the embedded refreshing portion remain the same across the refresh.
20 When LaunchTile transitions from World View to Zone View, it completely replaces content
21 displayed in the prior view with new content derived from separate underlying image resources.

22 111. For the reasons stated and shown above, LaunchTile does not disclose element [c]
23 of claim 50.

24 **D. Claim 50, Element [d]**

25 112. Neither does LaunchTile disclose the element of claim 50 of “instruction[s] for,
26 while the first box is enlarged, a second gesture is detected on a second box other than the first
27 box; and instructions for, in response to detecting the second gesture, the structured electronic
28

1 document is translated so that the second box is substantially centered on the touch screen
2 display.”

3 113. Mr. Gray opines that this element of claim 50 is disclosed by LaunchTile’s
4 launching of an application via a gesture on an application tile displayed in Zone View. (Gray
5 Decl. ¶¶ 92-95.) I disagree. Mr. Gray’s interpretation fails to appreciate the significance of the
6 act of launching an application in LaunchTile. This step, as LaunchTile’s name suggests, is of
7 critical—indeed, defining—importance to the system, and it distinguishes it fundamentally from
8 the invention of claim 50 of the ’163 patent.

9 114. As an initial matter, it is my opinion that claim 50’s requirement of a gesture on a
10 “second box other than the first box” is not met when the second box is wholly contained within
11 the first box. This is the case when, as Mr. Gray describes, a user enlarges a Zone View and then,
12 without scrolling to a different four-tile view, clicks on an application tile that is fully within the
13 Zone View that Mr. Gray defines as the “first box.” A “second box other than the first box”
14 requires, by its plain terms, a second box that is not any part of the first box.

15 115. The LaunchTile invention was, according to its authors, motivated by “the
16 problem [of] navigating device applications” on smartphone and PDA devices. (Bederson Decl.,
17 Ex. A at 201.) LaunchTile sought to provide, in particular, “high-value at-a-glance information for
18 several applications at once, as well as on-demand application launch when users desire more
19 detailed information.” (*Id.*) These dual goals central to LaunchTile’s design—(1) providing “at-
20 a-glance information” about several unlaunched applications simultaneously and (2) allowing
21 “on-demand application launch” to enable use of the applications’ full functionalities—
22 fundamentally define the way the system operates. Mr. Gray’s interpretation of LaunchTile,
23 however, ignores that it has the ability to “launch” applications at all, because he contends that
24 LaunchTile’s individual applications in their launched, interactive mode are part of the same
25 “electronic document” as the application-launching tiles displayed in the World View and Zone
26 View. This is contrary to the well understood meaning of launching an application, as well as the
27 treatment of applications in the Bederson paper on LaunchTile (Bederson Decl., Ex. A).

1 116. An application is a software program that performs particular user-oriented tasks.
2 When a user launches an application, such as the Email application, from LaunchTile’s Zone
3 View, that application fills the entire screen, and it allows the user to interact with the application
4 to perform the tasks that the application is designed to accomplish. In the case of LaunchTile’s
5 Email application, for example, launching the application displays a scrollable email client inbox,
6 and the user can select an individual email to open it. This application-level functionality has
7 nothing to do with the display of grids of application-launching tiles in the World View and Zone
8 View: the content is entirely different, and it supports entirely different user interaction for a
9 different purpose. It is not accurate, in my opinion, to say that the entirety of an interactive
10 application is part of the same “electronic document” as a set of application-launching tiles that
11 includes one tile that can launch that application. For example, when a user taps on an individual
12 LaunchTile, it is not “substantially centered” by “translating the structured electronic document”
13 as claim 50 requires—any more than a Microsoft Word icon on a Windows desktop is
14 “substantially centered” by “translating” the desktop’s array of application icons when the user
15 clicks on the icon to launch the Word application.

16 117. The Bederson paper on LaunchTile itself (Bederson Decl., Ex. A) repeatedly treats
17 the applications themselves as distinct from the World View and Zone View composed of
18 application-launching tiles. For example, the paper states that “Although the original focus of our
19 designs was on the application ‘shell’, we extended the LaunchTile interaction philosophy to the
20 application level, where we sought to make interaction consistent with navigation among the
21 application tiles.” (Bederson Decl., Ex. A at 205.) The LaunchTile authors distinguished
22 between “the application ‘shell’” that allowed “navigation among the application tiles” (i.e., the
23 World View and the Zone View) from the “application level” where interactive applications
24 themselves could be used. In describing implementation details, the paper again distinguishes
25 between the “shell” and “the applications themselves.” (*Id.* at 206.) The authors do not describe
26 “translating” any part of the shell to transition to the application level. Rather, they write that
27 “[t]he user taps any of the 4 notification tiles within Zone view to launch the corresponding
28 application.” (*Id.* at 205.) Thus it appears that even LaunchTile’s own authors did not conceive

1 of all three of LaunchTile’s levels as part of a unified electronic document. They treated the “the
2 application ‘shell’” (i.e., the World View and the Zone View)—which provided “high-value at-a-
3 glance information for several applications at once” (*id.* at 201)—as distinct from “the applications
4 themselves” (*id.* at 206) that gestures in the shell could “launch.”

5 118. My inspection and use of LaunchTile and XNav confirm that launching an
6 application from the Zone View does not merely “translate[e]” the representative content
7 displayed in the Zone View. For example, launching the email application brings up a more
8 detailed and longer list of emails in an inbox, and a user can select an individual email to view it.
9 Dr. Gray’s own images of LaunchTile’s transition at the launch of the email application show the
10 application’s different visual appearance from the static thumbnail provided in the Zone View:



11
12
13
14
15
16
17
18
19 119. The unimplemented applications in the LaunchTile and XNav prototypes—which
20 include 33 of the 36 notification tiles (all except the email application that Mr. Gray uses as his
21 example and two others)—provide the best illustration of the fact that the Zone View and the
22 Application View display entirely distinct content. Selecting a notification tile in Zone View that
23 corresponds to an unimplemented application results in the display of a placeholder screen—
24 shared by all of the unimplemented applications—that says “Application Under Construction.” It
25 is readily apparent that no notification tile in Zone View displays this text, so it cannot be the case
26 that an Application View displaying this text is the result of merely “translat[ing]” any electronic
27 document displayed in Zone View.
28



As shown above, selecting the unimplemented Calendar Application in XNav brings up a mostly blank screen with the text “Application Under Construction” on it.

120. Underscoring the separation between the Zone and World Views and the applications themselves, Mr. Gray conceded at his deposition that once a single application has been launched, there is no way to pan or scroll to see any other box of content from the World View or Zone View:

Q. And you cannot scroll or pan when you’re in the LaunchTile application view to see any of the adjacent LaunchTile zone view boxes, right?

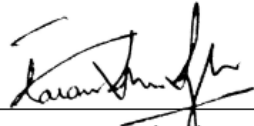
[Objection, compound]

A. My best recollection of the way that this operates is that—is that—let me think. I don’t, sitting here right now, I don’t remember certainly whether there is an ability to slide back to the other view. But I think not. I think—so let me—sorry.

I believe that from the selected second box, which has been expanded and centered on page 5, is labeled the “LaunchTile application view,” that it is—from there, I don’t know of a navigation path back to the first box other than to go back up to the world view and then select the zone again.

(Bartlett Decl. Ex. 30 at 214:16-215:6.) Mr. Gray’s testimony further confirms that the Application view is not obtained by merely “translating” or “scrolling” the structured electronic document from which the application tile was selected.

1 I declare under penalty of perjury under the laws of the United States of America that the
2 foregoing is true and correct. Executed on May 31, 2012.

3
4
5
6 
7 _____
8 Karan Singh

9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28