UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

KYOCERA CORPORATION
Petitioner

v.

SOFTVIEW LLC
Patent Owner,

Case IPR2013-00004
Patent 7,831,926

Before, SCOTT R. BOALICK, THOMAS L. GIANNETTI, and BRIAN J. McNAMARA, *Administrative Patent Judges*.

McNAMARA, *Administrative Patent Judge*.

DECISION
Institution of Inter Partes Review
37 C.F.R. § 42.108

BACKGROUND

Petitioner Kyocera Corporation requests *inter partes* review of claims 30, 31, 40, 41, 43, 52, 55, and 59 of US Patent 7,831,926 B2 (the ''926 patent) pursuant to 35 U.S.C. §§ 311 et seq. The Patent Owner, Softview LLC, did not file a preliminary response under 37 C.F.R. §42.107(b). We have jurisdiction under 35 U.S.C. §314.

The standard for instituting an *inter partes* review is set forth in 35 U.S.C. § 314(a) which provides as follows:

> THRESHOLD -- The Director may not authorize an inter partes review to be instituted unless the Director determines that the information presented in the petition filed under section 311 and any response filed under section 313 shows that there is a reasonable likelihood that the petitioner would prevail with respect to at least 1 of the claims challenged in the petition.

Petitioner challenges the claims as obvious under 35 U.S.C. § 103 on five separate grounds, designated in the Petition as RLP-1 through RLP-5. We grant the Petition challenging claims 30, 31, 40, 41, 43, 52, 55 and 59 based on the combination of Zarus, Pad++ and SVF (RLP-3) and the combination of Zarus, Hara, Tsutsuimatake, and SVG (RLP-5). We do not authorize *inter partes* review on all other grounds proposed by Petitioner, which are: the combination of Zarus and Pad++ (RLP-1); the combination of Zarus, Pad ++ and SVG (RLP-2); and the combination of Zarus, Pad++ and VML (RLP-4).

THE ‘926 PATENT (EXHIBIT 1001)

As indicated by its title, the ‘926 patent is drawn to the scalable display of Internet content, e.g., HTML based content, cascade style sheets (CSS) and XML, on mobile devices by enabling the content to be rendered, zoomed and panned for better viewing on small screens and standard monitors. (Col. 2, ll. 32-43, col. 5. Ll. 11-15). A client side viewer receiving the Internet content has an Internet browser and uses the simple vector format (SVF) originally designed to handle common computer-aided design (CAD) file formats to describe the current web content. (Col. 4, ll. 35-49). Translation of the content into a scalable vector representation can be done by a third party proxy service (Fig. 1A), the content provider’s web site (Fig. 1B), or at the client (Fig. 1C).

Figure 5 illustrates the logic used by the invention when translating content into a scalable vector representation. (Col. 3, ll. 40-42). Pre-rendering parsing of a received HTML document identifies elements such as tables, column definitions, graphic images, paragraphs and line breaks and determines where to place objects on a display. (Col. 15, ll. 45-52). When using frames, the display page is divided into multiple frame areas, which enables a single displayed page to include source code from several HTML documents. (Col. 15, ll. 33-36). During pre-rendering, each frame is examined in the sequential order it appears in the HTML document, and during further processing actual objects are rendered in their respective positions. (Col. 15, ll. 52-57). The content is separated into objects based on logical groupings of content and a page layout is built using bounding boxes produced for each object. (Col. 16, ll. 19-38, col. 17, ll. 15-29). The ‘926 patent admits that the above steps commonly are performed by conventional browsers in the pre-rendering process, but indicates that the steps it discloses to use layout data generated in the pre-rendering process to generate a scalable vector representation of the original page content depart from the prior art. (Col. 17, ll. 30-45).

The ‘926 patent discloses that generating a scalable vector representation begins by defining a page datum point as an X,Y value and a datum point as an X,Y value for each object’s bounding box. (Col. 17, ll. 45-64, col. 18, ll. 1-5). A vector between the page datum point and the datum point for each bounding box then is generated and stored. (*Id.*). A frame datum can also be assigned and vectors drawn from the page datum to the frame datum to establish the frame’s offset from the frame datum to each object in the frame. (Col. 18, ll. 5-16). The scalable vector representation is then completed by a reference that links each object’s contents, attributes such as type (image, text), and bounding box parameters, such as height and width, to the object’s vector. (Col. 18, ll. 18-26).

A display list of vectors for the vectorized HTML content is built, as is known from CAD arts, and a user-selectable scale and offset are determined. (Col. 19, ll. 14-25) The bounding boxes are processed using the scale and offset and a bounding box defining the limits of the display content is determined. (Col. 19, ll. 32-35). Scaling and offset can be accomplished by (i) mapping vectors to a virtual display area in memory with much more resolution than the actual display and reducing the scaling of the objects in the virtual display to how they will appear in the actual display or (ii) by using a fixed reference frame corresponding to the client's screen resolution and scaling and offsetting the vectors' bounding boxes relative to the fixed frame. (Col. 19, ll. 39-57). Using the latter approach, respective offsets in X and Y (-ΔX and -ΔY) are applied to the starting point and the vectors are scaled by an amount SF, producing a new datum (starting point) for each bounding box relative to the rendered page datum, which remains fixed but may or may not be displayed depending on the offset and scaling. (Col. 19, l. 58 – col. 20, l. 17). Once the bounding boxes are offset and scaled, the content (e.g., image and text) corresponding to objects having at least a part of their bounding boxes on the screen is retrieved from the client device's display list and scaled. (Col. 20, ll. 18 – 44). A display limit bounding box defines the portion of the display screen that actually will be used to display content. (Col. 19, l. 58 - col. 20, l. 7). The portions of the scaled content falling within the display limit bounding box are rendered on the client's display device. (Col. 20, ll. 45-47).

ILLUSTRATIVE CLAIMS

Claims 30 and 52, which are illustrative, are shown below with the paragraph designations used by Petitioner.

30. A mobile phone, comprising:

[A] a processor,

[B] wireless communications means operatively coupled to the processor, to facilitate communication with a mobile service provider network via which Web content may be accessed;

[C] a touch-sensitive display;

[D] a memory, operatively coupled to the processor; and

[E] storage means, operatively coupled to the processor, in which a plurality of instructions are stored that when executed by the processor enable the mobile phone to perform operations including,

[F] rendering a browser interface via which a user is enabled to request to access to a Web page having an original format comprising HTML-based content defining an original page layout, functionality, and design of content on the Web page;

[G] retrieving HTML-based content associated with the Web page;

[H] translating at least a portion of the HTML-based content from its original format to produce translated content including scalable vector-based content that supports a scalable resolution-independent representation of the HTML-based content that preserves an original page layout, functionality and design of the at least a portion of the HTML-based content when scaled and rendered; and

[I] employing the scalable vector-based content to render a view of at least a portion of the Web page on the display using a first scale factor,

wherein preservation of the functionality defined by the HTML-based content includes preservation of hyperlink functionality.

52. A mobile device comprising:

[A] a processor;

wireless communications means, to facilitate wireless communication with a network via which Web content may be accessed;

[B] a touch-sensitive display;

[C] flash memory, operatively coupled to the processor, in which a plurality of instructions are stored that when executed by the processor enable the mobile device to perform operations including,

[D] rendering a browser interface via which a user is enabled to request access to a Web page comprising HTMLbased Web content defining an original page layout, functionality, and design of content on the Web page;

[E] retrieving and processing the HTML-based Web content to produce scalable content; and

[F] employing the scalable content and/or data derived therefrom to, render a view of the Web page on the touch sensitive display; and

[G] re-render the Web page in response to associated user inputs to enable the user to iteratively zoom in and out views of the Web page while preserving an original page layout, functionality, and design defined by the HTML-based Web content as interpreted by a rendering engine,

wherein preservation of the functionality defined by the HTML-based Web content includes preservation of hyperlink functionality.

BASIS OF PETITION

Petitioner asserts there is a reasonable likelihood Petitioner will prevail on the following grounds:

GROUND	CLAIMS	BASIS
RLP-1 (Pet. pp. 21-34)[1]	30, 31, 40, 41, 43, 52, 55, 59, 72, 75	35 U.S.C § 103; Obvious over Zarus and Pad ++
RLP-2 (Pet. pp. 34-38)	30, 31, 40, 41, 43, 52, 55, 59, 72, 75	35 U.S.C § 103; Obvious over Zarus,Pad ++ and SVG

[1] All references to the Petition are to the Amended Petition filed on October 5, 2012 (Paper No. 6). Petitioner's Motion to Waive Petition Page Limit with an accompanying Amended Petition concurrently filed on October 5, 2012 (Paper No. 7) is denied.

RLP-3 (Pet. pp. 38-42)	30, 31, 40, 41, 43, 52, 55, 59, 72, 75	35 U.S.C § 103; Obvious over Zarus, Pad++ and SVF
RLP-4 (Pet. pp. 42-46)	30, 31, 40, 41, 43, 52, 55, 59, 72, 75	35 U.S.C § 103; Obvious over Zarus, Pad++ and VML
RLP-5 (Pet. pp. 46-58)	30, 31, 40, 41, 43, 52, 55, 59, 72, 75	35 U.S.C § 103; Obvious over Zarus, Hara, Tsutsumitake and SVG

ART CITED IN THE PETITION

Zarus (Exhibit 1004)

Petitioner identifies the Power Zarus as a touch screen Personal Digital Assistant (PDA) marketed in Japan by Sharp Corporation. The Petition cites six separate documents concerning the Power Zarus including a press release, its user manual, and several brochures. Petitioner cites Zarus to support contentions that accessing the Internet through either a wired or wireless connection with a browser that renders HTML pages and includes zooming functionality was known in the art. (Pet. 16).

Pad++ (Exhibit 1006)

Petitioner identifies Pad++ as a graphical interface system based on zooming that renders web pages for display in a resolution-independent, scalable format and enables zooming and panning while preserving the layout, design and function of a web page. (Pet. 17). Petitioner cites portions of eight separate Pad++ references published between 1995 and 1998, including five references by Bederson and others (Bederson 1 through Bederson 5), a Pad++ Reference Manual (Ref. Man.), a Pad ++ Programmer's Guide (Prog. Guide), and a Brief Tour of Pad++ by Jonathan Myer (Tour) (not included in Petitioner's Table of Exhibits). While the citation of eight documents as a single reference is unusual, each of the documents is tied closely to a single project, i.e., Pad++.

Pad++ provides a zooming user interface which manages rendering and interaction with graphical objects on a large information surface to present the user a zoomable view of the surface populated by graphical objects. (Bederson 4 pp. 1101, 1103 [Ex. 1006 pp. 117, 119]. A zooming user interface supports text, images, vector graphics, hierarchical groups and other objects to support navigation. (Bederson 4 p. 1104 [Ex. 1006 p. 120]. Users navigate using panning, zooming and hyperlinks over a large information surface on which documents can be placed and scaled to any size. (Bederson 4 p. 1101 [Ex. 1006 p. 117]). An object is a graphical entity manipulated by the system as a whole and may be a simple object, e.g., a line segment, or a compound object, e.g., an HML page. (*Id.*). Its developers assert that Pad++ is attractive for systems having small screens, such as hand held computers and PDAs. (Tour p. 141, [Ex. 1006 p. 341]).

Pad++ is a structured graphics widget for the Tcl/Tk scripting language and graphics extension that features zooming. (Prog. Guide p. 1, Bederson 5 pp.6, 24). Pad++ is not an application itself, but supports creation and manipulation of multi-scaled graphical objects. (Bederson 5 p. 6). The standard objects that Pad++ supports include text, graphics, images and HTML, such that Pad++ can read HTML and can follow links across the Internet. (Bederson 5 pp. 6, 11-12).

Pad++ maintains a distinction between a surface and a view. (Prog. Guide, p. 6). Every item on a pad surface exists at a specific place, and can be moved and scaled on the surface. (Prog. Guide. p. 4). A view specifies what portion of the pad surface is visible and at what magnification. (*Id.*). Thus, the appearance of an item on the screen depends upon the position and scale of the item (its transformation) as well as the location and magnification of the current view. (*Id.*). X and Y coordinate units, which relate to the Pad++ surface and are stored as floating point numbers, default to pixels, so that an item 50 pixels wide appears 100 pixels wide if the view has a magnification (zoom) of 2. (Prog. Guide pp.4-5). Coordinates may be specified as absolute or relative to something else, such as a specified frame or bounding box. (Prog. Guide p.5).

The view shows a given location of the surface specified by a list of 3 numbers representing (x,y) position and magnification, respectively. (Prog. Guide p. 6). The view is specified by the point currently at the center of the window and can be adjusted on the surface using the “moveto” widget command. (*Id.*). Graphical items, each with a unique ID, sit at a distinct location on a surface and have a given size. (*Id.*).

Every graphical item has an anchor position and anchor point associated with it that controls the item’s position and size, which can be accessed using the –anchor and “–place itemconfigure” options. (*Id.*). The “–place” option, which sets the anchor position of the object, consists of two values which specify the position of the anchor point, and a third value, which specifies its size. (*Id.,* Ref. Man. pp. 49-50). The “–anchor” option tells how to position the object relative to the position point specified by “–place” e.g., anchor position “center” results in the object being centered on the point, while “n” results in the object being drawn so that its top center point is at the positioning point. (Ref. Man. p. 47). Items with coordinates, such as rectangles, are created with -place option that is determined by the coordinates. (Prog. Guide p. 6).

An unlocked item can be moved or scaled relative to the surface using the –slide command, which shifts the item by adding *dx* and *dy* to the X and Y coordinates, and the –scale command, which multiplies the size of the item by the scaled amount around its center or a point (padX, padY) if specified in the command. (Prog. Guide, pp. 6, 7; Ref. Man. pp. 36, 38-39).

Zooming in on a view can be accomplished using the “moveto” command, which has parameters *xview, yview, zoom* that change the view so that the point *xview, yview* is at the center of the screen with a magnification of *zoom*. (Prog. Guide pp. 6,7; Ref. Man. p. 32). Except for ‘sticky” items, changing the view affects the way all items are rendered, while transforming a graphical item changes only the way that item is rendered. (Prog. Guide p. 7).

Pad++ allows specifying a bounding box, with a bbox command that returns a list of four elements (x1, y1, x2, y2) giving the bounding box for all the items named by the arguments of the command. The drawn area for all the named elements are within a region bounded by x1 on the left, x2 on the right, y1 on the bottom and y2 on the top. (Ref. Man. p. 17). Related commands include “center” which changes the view so as to center the first of the specified items so that the largest dimension of its bounding box fills a specified amount of the screen, and “centerbox,” which changes the view to center the specified bounding box so that its largest dimension fills the specified amount of the screen. (Ref. Man. p. 22).

<u>Scalable Vector Graphics (SVG) Requirements W3C Working Group Draft 29 Oct 1998 (Exhibit 1007)</u>

Petitioner cites the SVG Requirements as disclosing a family of specifications of an XML-based file format for two-dimensional vector graphics. (Pet. 18). According to Petitioner, SVG Requirements teaches scalable vector-based content, including zooming, panning, hyperlinks, HTML functionality, and Cascading Style Sheet (CSS). (*Id.*). Petitioner cites to the first page of SVG Requirements for the proposition that graphics in web documents will be displayable on a wider range of device resolutions from small mobile devices through office computer monitors to high-resolution printers and that this will be a significant advance in Web functionality. (Pet. 36).

<u>The Simple Vector Format (SVF) References (Exhibit 1009)</u>

Petitioner's Exhibit List identifies three SVF references (the SVF References) including a 1995 specification for the Simple Vector Format v. 1.1 (SVF), a March 1996 article entitled "Bring New CAD Viewing Power to the Internet" (SVF Press 2) and a June 1996 article entitled "New CAD System Works With AutoCAD Drawings Without Translation" (SVF Press 1). However, Exhibit 1009 appears to contain additional documents bundled with SVF including the Specification for SVF v. 2.0, SVF XML and Changes to SVF, and additional documents bundled with the documents identified in the Exhibit List as SVF Press 1 and SVF Press 2. Although Petitioner indicates that each of the SVF documents was published between 1995 and 1998, we note that the document "Changes to SVF" describes changes as late as Dec. 6, 2000. (p.1)

Petitioner cites the SVF References for disclosing a base set of coordinates (0,0) and describing vectors according to their endpoints (X,Y) in relation to that base set of coordinates. (Pet. 19). Petitioner also states that the SVF References disclose plug-ins that work with web browsers and include scalable vector graphics, permit zooming and panning of the displayed webpage, and preserve webpage functionality, such as hyperlinks. (*Id.*).

SVF purports to be a simple format for describing vector images that features layers for controlling the visibility of objects, hyperlinks for allowing the user to click on a certain portion of the drawing to perform an action, and notifications for sending messages when the user has passed a certain zoom level. (SVF pp. 1, 6-9). SVF designates the origin (0,0) in the lower left corner with increasing x coordinates to the right and increasing y coordinates moving up, such that coordinates are specified as positive integers, although offset values may be negative as well. (*Id.*). For example, in SVF, a line is drawn by specifying a starting point in X and Y coordinates and outputting a line to an ending point specified in X and Y coordinates. (SVF p. 2). The default extent (or size) for an image is (0,0) to (255,255), but because the size of an image can change, extent information can be set to allow an entire image to be displayed. (*Id.*). Text width can be specified and, if specified, scaled to fit. (SVF p.4).

SVF discloses that an image can contain miscellaneous information such as a suggested width, but that the display program is not required to honor the request. (SVF, p. 8). Among the miscellaneous information that can be added to an image is a scale and base point, but this information is not used for display. (SVF pp. 8-9). This information would be used if the original coordinate system needed to be recreated from the integer values stored in the file, using the SVFOutputTransform command. (*Id.*). In that case, an SVF coordinate pair would first be added to the base point, then each of the values would be multiplied by the scale factor. (*Id.*).

Vector Markup Language (Exhibit 1010)

The Vector Markup Language (VML) is an application of XML that defines a format for encoding vector information together with additional markup to describe how that information may be displayed and edited. (VML p. 1). The VML workflow generates locations and related information for vector paths and related objects, such as bitmaps, which are then rendered using native operating system functionality. (VML p. 3). In VML two dimensional coordinates are defined as single attributes. (VML pp. 7, 11). Vectors defining a shape can be specified in a local coordinate system. (VML p. 10). A shape element is used to define a visible vector graphic element, while a group element groups several shapes together so they can be transformed together as one unit. (VML pp. 8-9). The shape and group elements participate in the CSS2 visual rendering model in which positioning information is expressed in terms of a local coordinate space defined for any sub elements inside a containing block. (VML pp. 9-10). CSS2 position attributes (left, top, width, height, etc.) are numbers without a specific unit. (VML p. 10). If a containing block for a shape is changed, the outline of the shape will be automatically scaled to the new box. (*Id.*).

Hara (Exhibit 1008)

Recognizing that, because high resolution screens have a narrower dot pitch than low resolution screens, images on high resolution screens appear relatively smaller, Hara addresses modifying display size in accordance with the resolution of a display screen, to generate a display that can be recognized by a user. (Hara Abstract, ¶¶[0002,0003].). Hara describes a user's information processing device having a CPU, controller, program ROM, a RAM, a pointing device, display device and a VRAM for rendering data to be displayed by the display device. (Hara ¶¶[0023-0028]). The RAM stores a display table, an image size table, an image magnification parameter table, a user specified size table and a row information control table which store information to process the display. (Hara ¶¶[0031-0037]).

Hara discloses displaying information in accordance with the resolution of the display screen (Hara ¶¶[0045-0049]), displaying information in accordance with a user specified size (Hara ¶¶[0050-0052]), displaying a plurality of image data (Hara ¶¶[0053-0056]), displaying image data sent from a server to a client (Hara ¶¶[0057-0062]), displaying clickable data (Hara ¶¶[0063-0068]) and displaying information using a cache function (Hara ¶¶[0069]). For example, Hara (referred to as JP '169 in the Grimes 1 Declaration, Exhibit 1021) discloses that when an image is received in an HTML document, the resolution of the image is compared with the screen resolution to calculate the magnification that matches the width of the display screen. (Hara ¶¶[0058-0061]). If at this magnification the image overruns the display screen, then a magnification is calculated to match the vertical height of the display screen and the most appropriate magnification is selected. (*Id.*). The resolution of the image data is converted to a size that matches the display screen resolution by interpolating data between dots in the image data. (*Id.*). In describing the display of clickable data, Hara discloses that using the upper left of a window as the origin, a shape can be defined by X,Y coordinates, e.g., the range of a rectangle shape is defined by specifying the shape "rectangle" with coordinates (X1, Y1) and (X2, Y2), (*See*, Hara ¶ [0064], Fig. 9). At the time of display, a clickable control table stored in a RAM of the user's device stores the shape of the clickable region and its coordinates and information indicating the link destination. (Hara ¶[0066]). When a coordinate data event is produced through a pointing device, a determination is made concerning whether the event is within the coordinates controlled by the clickable control table and, if so, the file of the link destination is requested from the server. (Hara ¶[0067]). If there is a change in the display size of the image on the client (user device) side, the coordinate positions of the clickable data are shifted and the coordinate values for the clickables within the clickable control table are corrected based on the magnification stored in a magnification parameter table. (Hara ¶[0068]). Thus, if the client side resolution is such that the selected image data is magnified by 2x, the coordinates (X1, Y1) become (2xX1, 2x X2). (*Id.*, *See also,* Hara ¶¶[0091-0093]).

<u>Tsutsumitake (Exhibit 1005)</u>

Tsutsumitake discloses a device that receives and stores a document in an external format such as HTML, assigns the document an identifier, stores document state information and converts the document to an internal format suitable for display on a screen on the device, e.g., using information blocks, such that an X coordinate and a Y coordinate, to indicate the display position of each block. (Tsutsumitake¶¶[0010, 0025-0026], Figs. 2-5). When a document is to be displayed, the current scroll position is determined, the document state information is retrieved and the current scroll position is subtracted from the y coordinate value in the stored internal format, and the cursor is moved to display the document. (Tsutsumitake ¶¶[0035-0038], Fig. 6). In a second embodiment, Figure 10 of Tsutsumitake discloses that images can be displayed using summary display coordinate information stored in a table as shown in Figure 9. (Tsutsumitake¶¶[0015], [0048]). The summary coordinate information can include X and Y starting coordinates and the length of a line segment from those coordinates. (Tsutsumitaki Fig.9, ¶[0049]).

CLAIM CONSTRUCTION

Without admitting or acquiescing as to the correctness of the Patent Owner's claim construction proffered in the pending litigation, Petitioner states it would not be unreasonable to adopt the Patent Owner's proffered constructions for this proceeding. We have reviewed the Patent Owner's proposed constructions and determined that they are consistent with the broadest reasonable construction. Therefore, we adopt the following claim constructions:

CLAIM TERM	CONSTRUCTION
scalable content	graphic content capable of being rendered at multiple zoom levels
scalable/scaling/scaled	capable of being rendered at multiple zoom levels / rendering at multiple zoom levels / rendered at multiple zoom levels

processing the HTML-based Web content to produce scalable content	processing [the] HTML-based Web content to produce content capable of being zoomed in or out
vector based content	graphic content that includes one or more vectors
scalable vector-based content	graphic content that (1) is capable of being rendered at multiple zoom levels and (2) includes one or more vectors
vector	A mathematical expression representing a length and a direction in a two dimensional space. In an X,Y coordinate system, a vector is represented by a value X2,Y2 relative to an origin point, represented by X1,Y1
primary datum	an origin point defined at an X,Y coordinate
object datum	reference point for an object
layout location datum	one or more points corresponding to the location of the object
enabling the user to zoom and pan a view of the Web page	enabling the user to zoom and move around the Web page
machine-readable medium	The machine-readable medium may include, but is not limited to, floppy diskettes, optical disks, ROMs, RAMs, EPROMs, EEPROMs, magnetic or optical cards, flash memory, or other type of media/machine-readable medium suitable for storing electronic instructions

ANALYSIS OF THE CHALLENGES TO THE CLAIMS

The’926 patent states that in one embodiment, the client side viewer attains its small size and efficiency by taking advantage of the power of SVF (Simple Vector Format) to describe almost any current web content. (Col. 4, ll. 43-45). The

‘926 patent further notes that web content and functionality can be passed seamlessly to the end user platform without any degradation in the look or feel of the output and, because the file graphics are handled as vectors, the end user can control real time changes in the size of text and graphics, as well as what portion of the file is viewable, using “zoom and pan” capabilities familiar to CAD and other vector content users. (Col. 4, ll. 56-66).

The ‘926 patent also indicates that besides making small displays eminently usable, this technology enables web content to be displayed on normal desktops to assist the visually impaired. (Col. 5, ll. 9-15). This passage suggests that relevant references which teach zooming techniques are not limited to those which deal only with small devices and screens.

Claim 30

Claim 30 (and dependent claims 31, 40, 41 and 43) are drawn to a mobile phone. The Zarus device, which Petitioner cites as a reference in each of its challenges in the Petition, has a built-in modem (Zarus 1, p. 150 [Ex. 1004, p. 122]), but is not a mobile phone. The Zarus Pocket MI-100 series device is essentially a personal digital assistant (PDA) that can be equipped with a separately-sold digital cellular phone adapter or a PHS adapter for data communication, such as browsing the Internet. (Zarus 1, p. 19 [Ex. 1004, p. 94], Zarus2 p. 26 [Ex. 1004, p. 608], Zarus 2, p. 8 [Ex. 1004, p. 622]). *See also*, (Zarus2, p. 176 [Ex. 1004, p. 605], Zarus-2, p. 12-14 [Ex. 1004, p. 626-628]). The specification of the '926 patent notes that the client device may be a small device such as a hand held computer or a cell phone, which has a smaller display resolution than common web pages are designed for. (Col. 18, ll. 63-65). Therefore, although Zarus is not a mobile phone, and the relevant touch screen and display are on the PDA rather than the phone, we agree with Petitioner that the Zarus device and related documentation are relevant prior artfor purposes of determining whether claims drawn to a mobile phone are obvious. We further agree that the Zarus documentation discloses a processor, wireless communications means to facilitate communications with a web service provider to access web content (when used with the digital cellular phone adapter), a touch sensitive display (located on the Zarus PDA connected to the Internet through a digital cell phone), and a memory operatively connected to the processor. We also agree that the Zarus documentation shows the device includes a memory storing instructions that when executed allow the processor to render a browser interface (with some limitations) and retrieve HTML based content. (Zarus 2, p. 68, 69 [Ex. 1004, p. 638,639]). The Zarus documentation also indicates that the Zarus device provides horizontal and vertical scrolling (Zarus 1, p. 29 [Ex. 1004 p. 98]) and magnified and reduced views of web pages, but does not disclose how these features are implemented. (Zarus 2, p. 86,87 [Ex. 1004, p. 644, 645]). Thus, we agree that Zarus discloses or suggests elements [A]-[G] of claim 30 and demonstrates the desirability of using zooming and panning techniques to display HTML based content on mobile devices, such as mobile phones.

The documentation concerning Pad++ indicates its applicability to devices with small screens, such as PDAs like that disclosed by the Zarus documentation. (Tour p. 141 [Ex. 1006 p. 341]). As discussed above, the discussion in the ‘926 patent of the applicability of SVF and the usefulness of the same zooming techniques in large screen devices indicates a person of ordinary skill would look to zooming references generally in the prior art. (*See*, Grimes 1 Dec. ¶103-109 concerning the combination of Pad++ and SVF.).

The ‘926 patent admits that processing of HTML images was generally known and that the invention deviates from the prior art by using the various layout data generated during the pre-rendering process to generate a scalable vector representation of the original page layout. (Col. 17, ll. 43-45). This departure from the prior art appears to be the subject matter recited in elements [H] and [I] of claim 30.

Element [H] of claim 30 recites translating at least a portion of the HTML-based content from its original format to produce translated content including scalable vector–based content. (Claim 30). We have construed scalable vector-based content to be content that is capable of being rendered at multiple zoom levels and includes one or more vectors. We have construed a vector to be a mathematical expression representing a length and a direction in a two dimensional space that can be represented in an X,Y coordinate system by a value X2,Y2 relative of an origin point represented by X1,Y1. SVF describes drawing a line by specifying a starting point in X,Y coordinates and outputting a line to an ending point specified in X,Y coordinates (SVF, p.2). The Declaration of Jack D. Grimes, Ph.D. (Grimes 1 Declaration) supports the view that this approach in the Simple Vector Format is a vector representation. (Ex.1021 ¶¶ 94-96). Pad ++ states that one of its objectives is to support vector graphics. (Bederson-4, p. 1104 [Ex. 10006, p. 120]). Pad++ discloses representing a vector in an X,Y coordinate system using values X2,Y2 and an origin X1,Y1. (*See,* Pad ++ Prog. Guide, p. 4-5 [Ex. 1006, p. 186-187]). The Grimes 1 Declaration supports the view that specifying X,Y coordinates establishes a vector in 2 dimensional space as in Pad++. (Ex.1021, ¶¶ 23-56).

Pad++ discloses that a view specifies a portion of a surface with graphical items and its magnification and that a "moveto" command specifies a point *xview, yview* with a magnification of *zoom*. (Prog. Guide, pp. 6-7 [Ex. 1006, p. 188-189]). Pad++ further discloses that an item on a surface can be moved around using a *slide* command and scaled in size using a *scale* command. (Prog. Guide, pp. 6,7

[Ex. 1006 pp. 188-189]). The Grimes 1 Declaration states that these features permit zooming, panning and vector generation. (Ex. 1021, ¶¶ 57-68). In view of this evidence, we agree that Pad++ discloses scalable vector based content that supports a scalable resolution-independent representation of the HTML-based content.

The Grimes 1 Declaration also notes that SVF teaches scalable vectors and it discloses image scaling by adding a coordinate pair to a base point and multiplying each value by a scale. (Ex. 1021, ¶¶ 95-96).

Element [H] of claim 30 further requires that the scalable vector based content support preservation of the original page layout, functionality, and design of the HTML-based content when scaled and rendered. Pad++ can follow links across the Internet, preserving that functionality. (Bederson-5, p. 12, [Ex. 1006, p. 162]). The Grimes 1 Declaration states that the Bederson 3, Bederson 4 and Brief Tour documents demonstrate that Pad++ is capable of zooming and panning a single web page, while preserving the layout and functionality of the web page, including hyperlink functionality (as required by elements [H] and [I] of claim 30. (Grimes Decl. ¶¶113-115). In the absence of evidence to the contrary, we agree with Petitioner that Pad++ discloses the features recited in elements [H] and [I] of claim 30.

Given the admissions in the '926 patent concerning the SVF Reference and the evidence concerning Pad++, we agree that Petitioner has demonstrated a reasonable likelihood it will prevail with respect to obviousness of claim 30 based on Zarus, Pad++ and SVF.

Initiating trial on Zarus and Pad++ alone as requested in the Petition would be redundant because it would encompass all the prior art already included in other grounds asserted. We therefore do not authorize an *inter partes* review on the grounds challenging the patentability of claim 30 based upon Zarus and Pad++. 37 C.F.R. § 108(a).

SVG does not add to the disclosure in Pad++ and SVF. Therefore, we do not authorize an *inter partes* review on the grounds challenging the patentability of claim 30 based on the combination of Zarus and Pad++ and SVG.

Petitioner cites VML with respect to element [H] of claim 30. Petitioner cites VML as defining a format for encoding vector information in which a vector has numeric data in the form of 2D coordinates defined as single attributes (x,y) rather than pairs of attributes, with the a "coordorigin" attribute that defines the coordinate at the top left corner of a containing block. (Pet. 43-44). Petitioner cites no additional disclosure in VML that contributes information not already disclosed in Pad++ and SVF. Therefore, the combination of his Zarus, Pad++ and VML would be redundant to the combination of Zarus, Pad++ and SVF. We do not authorize an *inter* partes review on the grounds challenging claim 30 based on the combination of Zarus, Pad++ and VML

Petitioner cites Hara (Ex. 1008) as disclosing the display of clickable data in an HTML document. (Pet. 48). Hara discloses a client device receiving an HTML document from a server, analyzing it to determine whether there is image data to be displayed and processing the image for display and magnification depending upon the resolution of the client device. (Hara ¶¶[0058]- [0062]). In describing the display of clickable data Hara discloses using (x,y) coordinates and tables for shifting the display based on the magnification. (Hara ¶¶[0063]-[0068]). Referring to the image origin as (0,0), Hara discloses that in Figure 9(a), the range of one rectangle is defined by the coordinates (X1,Y1) and (X2,Y2), (Hara ¶[0064]). Because the origin of the window is (0,0), the point (X1,Y1) defines a vector from the origin. (*See*, Grimes 1 Decl., Ex. 1021, ¶¶[0087- 0092]). Because the vector can be magnified, the display is scalable. (*Id.*).

Petitioner further cites Tsutsumitake (Ex. 1005). Petitioner notes that Tsutsumitake discloses receiving an HTML document, translating the HTML to (x,y) coordinate information for text and images to be displayed on a client device and storing the information in a table. (Pet. 49). Referencing Fig. 9, Petitioner notes that Tsutsumitake discloses that straight lines can be specified by starting coordinates and a length, indicating a vector representation. (Pet. 49). Petitioner further sites the SVG requirements document as indicative of the applicability of scalar vector graphics to the World Wide Web.

In view of the disclosure in Zarus, Hara, Tsutsumitake and SVG, we are persuaded that Petitioner has established a reasonable likelihood of prevailing and we therefore grant the Petition challenging claim 30 based on Zarus, Hara, Tsutsumitake and SVG.

Claim 31

Having not authorized an *inter partes* review on the grounds challenging independent claim 30 based on Zarus and Pad ++, Zarus, Pad++, and SVG, and Zarus, Pad++ and VML, we do not authorize an *inter partes* review of dependent claim 31 based on these same combinations of references.

Claim 31 recites the further limitation on claim 30 that execution of the instructions performs operations allowing the user to zoom on a user selectable portion of a display using a touch sensitive display. Because Zarus discloses zooming using a touch sensitive device, we are persuaded that Petitioner has established a reasonable likelihood of prevailing and we grant the Petition challenging claim 31 based on the combination of Zarus, Pad++ and SVF and on the combination of Zarus, Hara, Tsutsumitake and SVG.

Claim 40

Having not authorized an *inter partes* review on the grounds challenging independent claim 30 based on Zarus and Pad ++; Zarus, Pad++, and SVG; and Zarus, Pad++ and VML, we do not authorize an *inter partes* review of dependent claim 40 based on these same combinations of references.

Claim 40 recites that execution of the operational instructions allow the user to view a column of web content at a higher resolution than the current resolution by tapping the column via the touch sensitive display and re-rendering the display such that the content corresponding to the selected column is displayed to fit across the touch sensitive display. As previously noted, Zarus discloses selecting content with a touch sensitive display. Pad++ includes a renderer that performs all rendering to the screen, maintaining a stack of transformations, including separate stacks of view transformations and object transformations, that specify translations and scale. (Bederson 4, pp. 1128, 1132 [Ex. 1006, pp. 144, 148]). Pad++ discloses a bounding box (Ref Guide, p. 17, [Ex 1006 p. 19]) and commands such as "center' and "centerbox" to center and scale items to fill a part of the screen (Ref. Guide, p. 22 [Ex. 1006, p. 24]) as well as the ability to specify the width of an item (Ref. Guide, p. 2 [Ex. 1006, p. 4]. Pad++ also provides for manipulating text items, which displays a string of characters on the screen in one or more lines. (Ref. Guide, pp. 39-40, 75-77 [Ex. 1006, pp. 41-2, 77-79]). SVF discloses that width can be specified with text and that the text will be scaled to fit the width. (Spec v. 1.1., p. 4, Spec. v. 2.0, p. 8 [Exc. 1009, pp. 6,19]). In view of these disclosures, we are persuaded that Petitioner has demonstrated a reasonable likelihood of prevailing and we grant the Petition challenging claim 40 on the basis of Zarus, Pad++ and SVF.

In discussing the display of clickable data, Hara discloses that an area shape, such as a rectangle (which could include a column of web content, as recited in claim 40) can be defined by x,y coordinates and scaled for display on the target device (Hara ¶¶[0064]- [0068]) and that the magnification can be set to match the width of the display screen. (Hara ¶[0060]). In view of these disclosures, we are persuaded that Petitioner has demonstrated a reasonable likelihood of prevailing and we grant the Petition challenging claim 40 based on Zarus, Hara, Tsutsumitake and SVG.

Claim 41

Claim 41 recites that the web content contains at least one image and that execution of the instructions performs further operations enabling a user to view an image at higher resolution by tapping on the image such that the display is re-rendered for the image to fit across at least one of a width and height of a display area of the touch sensitive display. As discussed above, Pad++ provides bounding boxes containing images and commands that allow the images in the bounding boxes to be expanded so the largest dimension fills the specified amount of the screen. (Ref. Guide, p. 22 [Ex. 1006, p. 24]). In view of these disclosures, we are persuaded that Petitioner has demonstrated a reasonable likelihood of prevailing and we grant the Petition challenging claim 41 based on Zarus, Pad++ and SVF.

Hara in Fig. 8(c) discloses an image selected by a pointing device that is scaled to fit across the width and height of the screen. Therefore, we are persuaded that Petitioner has demonstrated a reasonable likelihood pf prevailing and we grant the Petition challenging claim 41 based on Zarus, Hara, Tsutsumitake and SVG.

Having not authorized an *inter partes* review of independent claim 30 based on Zarus and Pad ++; Zarus, Pad++, and SVG; and Zarus, Pad++ and, we do not authorize an *inter partes* review of dependent claim 41 based on these same combinations of references.

Claim 43

Claim 43 recites generating a display list associated with the scalable vector content and employing the display list to re-render the display at different scale factors to enable rapid zooming of the Web page. The '926 patent discloses that vector representative data for a web page is gathered and stored in a cache at the client from which it is retrieved. (Col. 19, ll 14-19). The '926 patent next states that a display list of vectors is built and that this process is well known in the CAD arts and enables the rapid zooming of vector based objects. (Col. 19, ll. 20-23). Pad ++ discloses that it is built using X windows, a 2D graphics system, which makes heavy use of caching. (Bederson-4, p. 1108 [Ex. 1006, p. 124]). As previously discussed Pad++ places graphical objects using (x,y) coordinates on surfaces and maps surfaces to the screen through views, which specify the position and magnification at which the associated surface is perceived, so that the extent of the surface that is seen is indirectly specified by windows and portals associated with views. (*See*, Bederson-4, pp. 1128- 1130 [Ex. 1006, p. 144 – 146]). Pad++ provides an example of creating a surface, putting in a top level window, and creating several objects and changing the view. (Bederson4, pp.1130-1131 [Ex. 1006, pp. 147-147]). Therefore, we are persuaded that Petitioner has demonstrated a reasonable likelihood of prevailing and we grant the Petition challenging the patentability of claim 43 on the basis of Zarus, Pad++ and SVF.

Tsutsumitake shows a display list (*See*, Fig. 3) specifying the position of images and text to be displayed using (x,y) coordinates, as well as scrolling so the information can be re-rendered at a desired position (Tsutsumitake ¶¶[0038-0039]). Hara discloses a RAM 12 for temporarily storing received data, such as HTML data, and a VRAM 17 for rendering data to be displayed by the display device after resolution conversion. (Hara ¶¶[0028-0029, 0058-0061]). Hara also discloses using (x,y) coordinates to the establish location of the clickable image data resulting in re-rendering the image by shifting of the image to accommodate the resolution of the display. (Hara ¶¶[0063-0068]). Hara achieves this using a display table, an image size table, a magnification parameter table and other tables. (Hara ¶[0031]). Therefore, we are persuaded that Petitioner has demonstrated a reasonable likelihood of prevailing and we grant the Petition challenging the patentability of claim 43 based on the combination of Zarus, Hara, Tsutsumitake and SVG.

Having not authorized an *inter partes* review of independent claim 30 based on Zarus and Pad ++; Zarus, Pad++, and SVG; and Zarus, Pad++ and VML, we do not authorize an *inter partes* review of dependent claim 43 based on these same combinations of references.

Claim 52, 55, and 59

Independent claim 52, which is drawn to a mobile device, rather than a mobile phone, recites many of the device features found independent claim 30. Unlike claim 30, which recites translated content includes scalable vector based content, claim 52 recites processing HTML based web content to produce scalable content that is employed to render a view of the web page on a touch sensitive display and re-render the web page in response to user inputs to enable the user to iteratively zoom in and out views of the web page, while preserving web page layout, design and functionality, including hyperlink functionality. Zarus shows iterative zooming in a touch screen mobile device, while Pad++ discloses iterative zooming, for example in Bederson 5, zooming from a, to b to c or Bederson 3, Figures 5, 6, and 7 and Tour. (*See*, Bederson5, p. 17, [Ex. 1006, p. 167], Bederson 3, p. 265 [Ex. 1006, p. 109], Tour pp. 87-89[Ex. 1006, pp., *See also*, Grimes 1 Decl. ¶¶77-78 [Ex. 1021, p. 13]). The remaining features of claim 52, including receiving HTML and preserving layout and hyperlink functionality were discussed relative to claim 30. Therefore, we are persuaded that Petitioner has demonstrated a reasonable likelihood of prevailing and we grant the Petition challenging the patentability of claim 52 based on the combination of Zarus, Pad++ and SVF.

Similarly, as discussed with respect to claim 30, Hara and Tsutsumitake show vector based zooming and translation of HTML. Zarus shows zooming iteratively. We therefore grant the Petition challenging the patentability of claim 52 on the basis of the combination of Zarus, Hara, Tsutsumitake, and SVG.

Like claim 31, claim 55 adds the further limitation that executing the instructions enables the user to zoom in on a user selectable portion of the display in response to a user input via the touch sensitive display. For the same reasons we grant the Petition with respect to claim 31, we grant the Petition with respect to claim 55 on the same grounds, i.e., the combination of Zarus, Pad++ and SVF and the combination of Zarus, Hara, Tsutsumitake and SVG.

Dependent claim 59 adds the additional limitation that the scalable content be vector based content, which is not a requirement of independent claim 52. For reasons discussed above with respect to claim 30 Petitioner has established a reasonable likelihood of success in demonstrating that the content is vector based. Therefore, we grant the Petition challenging the patentability of claim 59 based on the combination of Zarus, Pad++, and SVF and the combination of Zarus, Hara, Tsutsumitake and SVG.

We do not authorize an *inter* partes review of claims 52, 55 and 59 based on the combination of Zarus and Pad++; Zarus, Pad++, and SVG; and Zarus, Pad++, and VML for the same reasons we did not authorize an *inter partes* review of claim 30 on these grounds.

Claim 72

Claim 72 which depends from claim 52 recites the added limitations of [A] parsing the HTML code and [B] logically grouping selected content into objects, [C] defining a primary datum corresponding to the original page layout, and [D] for each object defining an object datum that corresponds to a layout location datum for the object's associated display content, [E] generating a vector from the primary datum to the object datum, and [F] creating a reference that links the object to its corresponding vector.

The specification of the '926 patent states that parsing HTML content to identify layout information, separating that content into logical objects and defining a bounding box for each object (steps 150, 152, and 154 in Fig. 5) are functions that are commonly performed by conventional browsers (Col. 17, ll. 31-34). Pad++ also discloses objects stored internally based on a hierarchy of bounding boxes. (Pet. 32, Bederson5 p. 23 [Ex. 1006, p. 173]).

We have construed the term "primary datum" to mean "an origin point defined by an X,Y coordinate" and the term "object datum" to mean "a reference point for an object." Claim 72 recites that the primary datum is for the original page layout, meaning an (x,y) coordinate for the page. The specification states that any point on the page may be used as the page datum, the only requirement being that the selected point be used consistently, and that the upper left corner of the display frame may be an advantageous selection. (Col. 17, ll. 50-55). Similarly, the '926 patent discloses that an object datum point may be anywhere on the object, such as the upper left corner of the bounding box, as long as that point is used in a predictable manner. (Col. 17, ll. 57-64). The '926 patent discloses that the vector for a given object may be stored as the X,Y value of the datum point of the object relative to (0,0), where the page datum point corresponds to the upper left corner of the display frame is assigned a value (0,0). (Col. 18, ll. 1-5). As previously discussed, Pad++ employs an X,Y coordinate frame with objects existing on surfaces and views defining the portion of the surface that is visible. In Pad ++, graphical items sit at a distinct location on a surface. (Prog. Guide, p. 6 [Ex. 10006, p. 186]). Initially the view onto the Pad++ surface looks at the origin with a magnification of 1.0, and is always represented by 3 numbers, i.e., x, y and magnification. (*Id.*). Each item has an anchor point associated with it that controls the item's position and size, as specified by "-place" where the first two values specify the (x,y) values and the third specifies its size. Thus, Pad++ discloses specifying a vector by the x,y value of the datum point of the object relative to a primary datum point. Each object in Pad++ has a unique id, linking the object to its vector. (Bederson5, p.24 [Ex. 1006, p. 174].

Pad++ discloses "Group Items" which provide a structure for grouping other items which may include HTML item types. (Ref. Man. p. 59 [Ex. 1006, p. 61]). Modifying the position of a group affects all the members, which are rendered sequentially in the display list. (*Id.*). Groups automatically resize themselves to include all of their members. (*Id.*). Pad++ further discloses that when HTML data is fetched, it is parsed and an HTML item is created which contains a method of rendering the page. (Ref. Man. p. 60 [Ex. 1006, p. 62]). HTML anchors are created as separate items and are automatically grouped with the HTML object

(*Id.*, Ref. Man. p. 63 [Ex. 1006, p. 65]). An HTML item is a group and may contain other members in addition to its anchors, which allows setting and retrieving all members that are part of the HTML item. (Ref. Man. p. 62 [Ex. 1006, p. 64]). Thus, Pad++ also discloses parsing HTML and logically grouping selected content into objects.

In view of these disclosures, we grant the Petition challenging claim 72 on the basis of Zarus, Pad++ and SVF.

Tsutsumitake discloses converting HTML documents (*See*, Fig. 2) into a form appropriate for a display device by using (x,y) coordinates to specify the positions of uniquely identified portions of the HTML document relative to an origin and storing a logically grouped list of such items to be displayed. (Figs. 3-5, 9-10). Hara discloses identifying images, for example, two images on the same line, and calculating the magnification (width and height) to display each of the images on the target display screen, based on width and height information stored in a row information table. (Hara ¶¶[0087-0088]). Hara also uses (x,y) coordinate data and discloses shifting coordinate locations for clickable data identified by a unique link when the display size of the image is changed on the display side. (Hara ¶¶[0063-0068, 0092). In view of these disclosures, we are persuaded that Petitioner has demonstrated a reasonable likelihood of prevailing and we grant the

Petition challenging claim 72 based on the combination of Zarus, Hara, Tsutsumitake and SVG.

Having not authorized an *inter partes* review of independent claim 52 based on Zarus and Pad ++; Zarus, Pad++, and SVG; and Zarus, Pad++ and VML, we do not authorize an *inter partes* review of dependent claim 72 based on these same combinations of references.

Claim 75

Claim 75 recites that the original format of the web page defines a width for the web page and that the execution of the instructions performs operations to determine a scale factor to fit the web page width across a display area of the touch sensitive device and employs the scale factor to render the display.

Pad++ discloses that dynamic objects can be placed at any position and at any scale (Bederson3, p. 262 [Ex. 1006, p.106]). Pad++ discloses that one can specify width in pixels (Ref. Man. p. 13 [Ex1006, p. 15]) and that an X server typically provides the screen width (Ref. Man. p. 14 [Ex. 1006, p. 16]). In addition, Pad ++ discloses that when a window is specified, all coordinates are scaled to fit the existing width of the window. (Ref. Man. p. 45, [Ex. 1006, p. 47]).

Pad ++ discloses fetching and parsing HTML data and creating an HTML item which contains a method for rendering the page. (Ref. Man. p.60 [Ex 1006,

p. 62]). Pad ++ also discloses the option of using “-width width” to specify the width in current units of the HTML page, so that the page will be re-laid out according to a new width, with the length of the page changing depending upon the new width. (Ref. Man., p. 63 [Ex. 1006, p. 65]). In view of these disclosures, we grant the Petition challenging claim 75 on the basis of the combination of Zarus, Pad++ and SVF.

As previously discussed, Hara discloses receiving HTML data and storing it in a RAM, calculating a magnification that matches the width of the display screen, and after resolution conversion placing the image data into VRAM to be displayed. (Hara ¶¶[0057]0061]. In view of this disclosure, we are persuaded that Petitioner has demonstrated a reasonable likelihood of prevailing and we grant the Petition challenging claim 75 based on the combination of Zarus, Hara, Tsutsumitake, and SVG. (RLP-5).

Having not authorized an *inter partes* review of independent claim 52 based on Zarus and Pad ++; Zarus, Pad++, and SVG; and Zarus, Pad++ and VML, we do not authorize an *inter partes* review of dependent claim 75 based on these same combinations of references.

SUMMARY

I. The Petition is GRANTED as to the following grounds asserted under 35 U.S.C. § 103:

Claims 30, 31, 40, 41, 43, 52, 55, 59, 72, and 75 as obvious over the combination of Zarus, Pad++ and SVF ; and

Claims 30, 31, 40, 41, 43, 52, 55, 59, 72, and 75 as obvious over the combination of Zarus, Hara, Tsutsumitake, and SVG.

II. We do not authorize an *inter partes* review on the following grounds asserted under 35 U.S.C. § 103:

Claims 30, 31, 40, 41, 43, 52, 55, 59, 72, and 75 as obvious over the combination of Zarus and Pad++;

Claims 30, 31, 40, 41, 43, 52, 55, 59, 72, and 75 as obvious over the combination of Zarus, Pad++ and SVG; and

Claims 30, 31, 40, 41, 43, 52, 55, 59, 72, and 75 as obvious over the combination of Zarus, Pad++ and VML.

ORDER

In consideration of the foregoing, it is hereby:

ORDERED that the Petition is granted as to claims 30, 31, 40, 41, 43, 52, 55, 59, 72, and 75 of the '269 patent.

FURTHER ORDERED that pursuant to 35 U.S.C. § 314(a) an *inter partes* review of the '926 patent is hereby instituted, commencing on the entry date of this Order, and pursuant to 35 U.S.C. § 314(c) and 37 C.F.R. § 42.4, notice is hereby given of the institution of a trial.

FURTHER ORDRED that the trial is limited to the grounds identified in Section I. of the above Summary, and no other grounds are authorized.

FURTHER ORDERED than an initial conference call with the Board is scheduled for 3 PM Eastern Time on May 1, 2013. The parties are directed to the Office Trial Practice Guide, 77 Fed. Reg. 48756, 48765-66 (Aug. 14, 2012) for guidance in preparing for the initial conference call, and should come prepared to discuss any proposed changes to the scheduling order entered herewith and any motions the parties anticipate filing during the trial.

PETITIONER:

Richard Bauer
Michael Tomsa
Katten Muchin Rosenman, LLP
Email: Richard.bauer@kattenlaw.com
Email: Michael.tomsa@kattenlaw.com

PATENT OWNER

Ben Yorks
Babak Redjaian
Irell & Manella, LLP
Email: byorks@irell.com
Email: bredjaian@irell.com