Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2609992 A1 2006/12/14

(21) 2 609 992

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(86) Date de dépôt PCT/PCT Filing Date: 2006/06/06
(87) Date publication PCT/PCT Publication Date: 2006/12/14
(85) Entrée phase nationale/National Entry: 2007/11/27
(86) N° demande PCT/PCT Application No.: US 2006/021767
(87) N° publication PCT/PCT Publication No.: 2006/133105
(30) Priorités/Priorities: 2005/06/06 (US60/688,274);
2006/01/26 (US11/341,231)

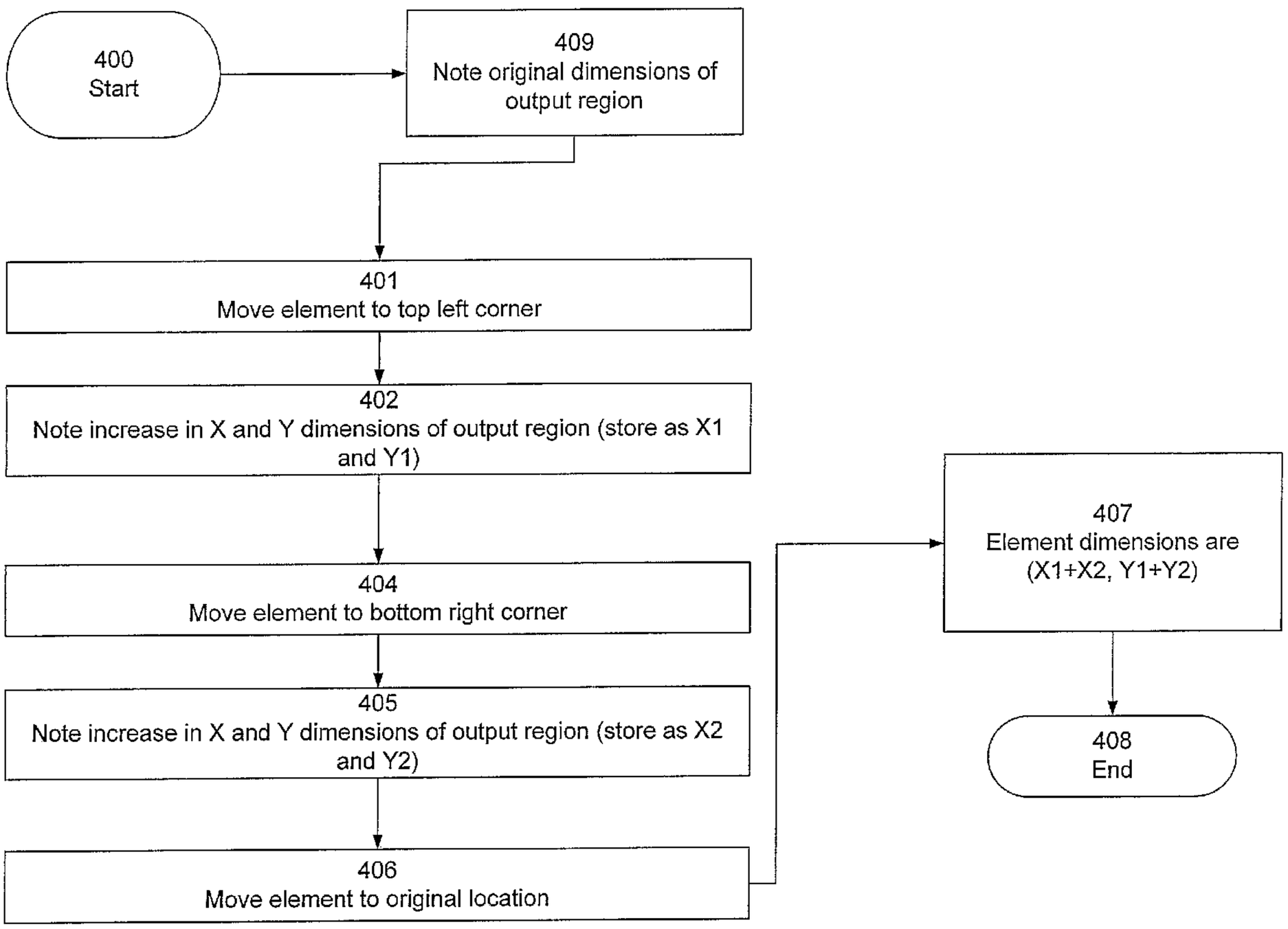
(51) Cl.Int./Int.Cl. *G06F 3/00* (2006.01)

(71) Demandeur/Applicant:
OMNITURE, INC., US

(72) Inventeurs/Inventors:
BAILEY, MICHAEL PAUL, US;
ERROR, BRETT M., US

(74) Agent: RIDOUT & MAYBEE LLP

(54) Titre : RECOUVREMENTS D'USAGE DU WEB POUR CONTENU DE PLUGICIEL DU WEB D'UN TIERS
(54) Title: WEB USAGE OVERLAYS FOR THIRD-PARTY WEB PLUG-IN CONTENT



(57) Abrégé/Abstract:
Overlay reports showing user interaction with web plug-in content are generated. Dimensions of elements within plug-in resources are determined by moving the elements to various locations with an output region and noting changes in overall dimensions of the out put region. Once dimensions have been determined, overlay reports are generated including color-coded regions depicting relative levels of interaction.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
14 December 2006 (14.12.2006)

PCT

(10) International Publication Number
WO 2006/133105 A2

(51) International Patent Classification:
G06F 3/00 (2006.01)

(21) International Application Number:
PCT/US2006/021767

(22) International Filing Date: 6 June 2006 (06.06.2006)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

60/688,274	6 June 2005 (06.06.2005)	US
11/341,231	26 January 2006 (26.01.2006)	US

(63) Related by continuation (CON) or continuation-in-part (CIP) to earlier application:

US	10/794,809 (CIP)
Filed on	3 March 2004 (03.03.2004)

(71) Applicant *(for all designated States except US)*: **OMNITURE, INC.** [US/US]; 550 E. Timpanogos Circle, Orem, UT 84097 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **BAILEY, Michael, Paul** [US/US]; 677 E. 300 North #1, Provo, UT 84606 (US). **ERROR, Brett, M.** [US/US]; 709 E. Sunrise Drive, Orem, UT 84097 (US).

(74) Agents: **RAUBVOGEL, Amir, H.** et al.; Fenwick & West LLP, Silicon Valley Center, 801 Califordnia Street, Mountain View, CA 94041 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US (patent), UZ, VC, VN, YU, ZA, ZM, ZW.

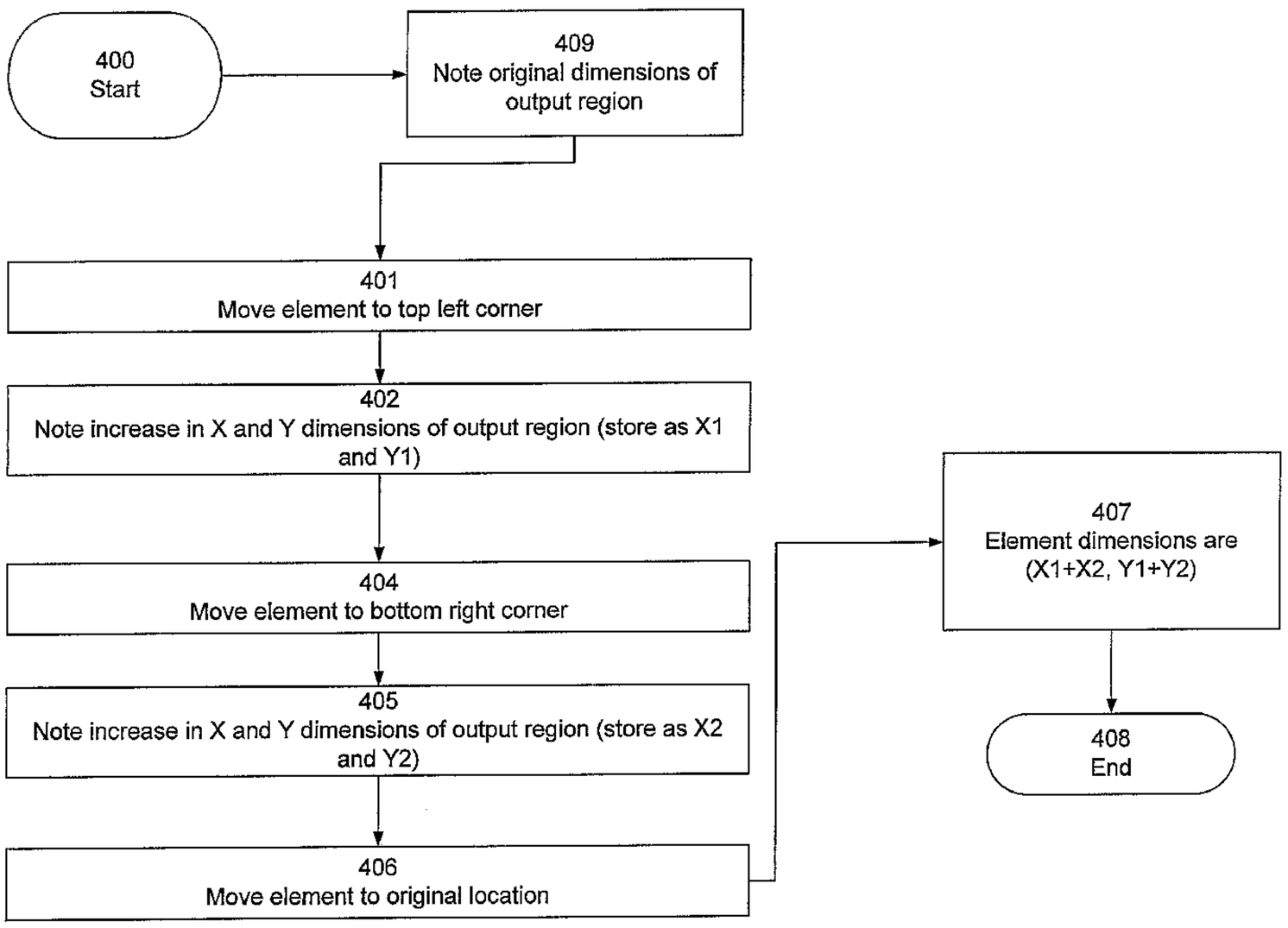
(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report*

[Continued on next page]

(54) Title: WEB USAGE OVERLAYS FOR THIRD-PARTY WEB PLUG-IN CONTENT



(57) Abstract: Overlay reports showing user interaction with web plug-in content are generated. Dimensions of elements within plug-in resources are determined by moving the elements to various locations with an output region and noting changes in overall dimensions of the out put region. Once dimensions have been determined, overlay reports are generated including color-coded regions depicting relative levels of interaction.

WO 2006/133105 A2

WO 2006/133105 A2

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WEB USAGE OVERLAYS FOR THIRD-PARTY WEB PLUG-IN CONTENT

Inventors:

Michael Paul Bailey

Brett M. Error

Cross-Reference to Related Applications

[0001] The present application claims priority from U.S. Provisional patent application serial no. 60/688,274, for "Applying A Web Usage Overlay To Output Of Third-Party Web Plug-Ins," filed June 6, 2005 and U.S. patent application serial no. 11/341,231, for "Web Usage Overlays for Third-Party Web Plug-In Content," filed January 26, 2006, the disclosures of which are incorporated herein by reference.

[0002] The present invention is a continuation-in-part of U.S. Utility patent application serial no. 10/794,809, for "Associating Website Clicks with Links on a Web Page," filed March 3, 2004, attorney docket number 7762, the disclosure of which is incorporated herein by reference.

Field of the Invention

[0003] The present invention relates generally to tracking website usage, and more particularly to monitoring and tracking user interaction with third-party web plug-ins and related content.

Background of the Invention

[0004] Web analytics refers to the analysis of data created by website usage. For instance, web analytics can be used to mine visitor traffic data. A variety of visitor traffic data is measured such as what browser is being used, what links on a given web page were selected, whether a product was purchase, etc. A number of web analytics tools are presently available, such as Site Catalyst version 11 from Omniture of Orem, Utah. These tools are able to capture data on website usage, and responsive to a user's request display a variety of different metrics on website usage such fallout/conversion, A/B testing, and the like.

[0005] In an on-line environment, website usage and other customer behavior may be tracked by a website server, or by another server such as a data collection server (also known

as a data collector), which may be remotely located. The data collection server is notified of activity on a website so that it can monitor and track the activity. One method of achieving this notification is through the use of a request for embedded content.

[0006] Embedded content is part of a web page, such as an image, that is requested as a separate file from the file containing the web page. The separate file may be requested from the website server or from a remote server, such as a remote content server or data collection server. For example, when a user requests a web page from a website server, the website server sends the web page file to the user's client. The client, such as a web browser, then attempts to render the file as a viewable web page. However, upon rendering the web page file, the client may find a reference to a separate file located on the website server or a remote server. After the content is located and sent to the client, the client renders the separate file containing the embedded content along with the original web page.

[0007] A web beacon (also known as a web bug) is a particular type of embedded content where the content itself is irrelevant, but the request for content carries useful information. For example, a web beacon is often a transparent image having very small dimensions, such as 1 pixel by 1 pixel. This image is small enough to be invisible to the user. When a client is rendering a web page that includes a web beacon, the web beacon causes the client to send a resource request to a server such as a data collection server. The web beacon may include a script (or other code) that causes the client to include, in the resource request, additional information about the user and the user's environment. The additional information can include the data from a cookie, or other information about the client's operating environment or status. Where the server indicated by the web beacon code is a data collection server, the data collection server may, in response to the request, cause the client to set an additional cookie for identification for tracking purposes. In this manner, the web beacon request can be used to indicate to a data collection server that a particular web page is being rendered.

[0008] One method for including the request is to write the request as a static image tag in Hypertext Markup Language (HTML). The following is an example of an image tag in HTML:

[0009] <img src="http://example.com/tracker.exe?AID=14658&PID=259294&banner=0.gif" width=1 height=1 border=0>

[0010] Here, the term "ad.datacollectionserver.com" refers to the address of the data collection server.

[0011] Another common method of including the request is to use a scripting language such as JavaScript so as to cause the browser to dynamically generate a request to the data collection server. One advantage of using a script instead of a static image tag is that the script can cause the browser to perform other functions including gathering additional data and sending it along with the request. In either case, the result is a request sent to the data collection server upon the occurrence of an event, such as the loading and rendering of a web page.

[0012] Once the request has been sent to the data collection server, the data collection server can perform various types of tracking functions. For example, the data collection server can count the number of requests associated with a web page so as to monitor traffic on the web page. By counting the number of times the web beacon element has been requested from the data collection server, the server can determine the number of times a particular page was viewed. By using JavaScript to dynamically construct the request for the web beacon and encode additional information, other identifying information can be obtained for further analysis.

[0013] Other types of website usage tracking are also well known, such as for example log file analysis. In such an approach, statistical analysis is performed on server logs in order to detect and analyze website traffic and usage patterns.

[0014] One mechanism for presenting website usage and traffic data, as described in related U.S. Utility patent application serial no. 10/794,809, for "Associating Website Clicks with Links on a Web Page," filed March 3, 2004, is to superimpose color-coded indications on a representation of a web page (or other online resource). Different colors can be used to represent different levels of activity for the various portions of the web page. For example, on-screen links that experience heavy traffic can be overlaid with one color, while links that experience less traffic can be overlaid with another color. A legend can be provided for explanatory purposes with respect to each of the colors.

[0015] For example, referring now to Fig. 2, there is shown an example of a page analysis report 201, displayed alongside an image of the web page 102 being analyzed. In

one embodiment, report 201 is provided to a site administrator or owner interacting with data collection server 106.

[0016] In the example of Fig. 2, report 201 includes identification 202 of the website and web page being analyzed; report date 203; report options and settings 204; page metrics 205; and links 206 to related reports. In addition, variable levels and shades of color density are superimposed on the displayed view of web page 102, in order to visually represent the relative number of clicks each item 208 or screen region has received. Color key 207 is a legend to indicate the meaning of various superimposed colors.

[0017] It is desirable to generate web usage overlays to areas of a web page that are occupied by or rendered by web plug-ins, including plug-ins provided by a third party, so as to provide a graphical display of user behavior with respect to content provided by such plug-ins. For example, it may be desirable to display a color-coded overlay showing user interactions with various elements of an interactive Flash movie, Java applet, or the like. In order to display an overlay having visual characteristics that align with the underlying elements (for example, to color-code various user interface elements), it is necessary to ascertain the dimensions of each of the underlying elements.

[0018] What is needed, therefore, is a technique for determining the dimensions of elements within plug-in content, even when such content was provided by a third party. What is further needed is a technique for generating a web usage overlay for such plug-in content.

Summary of the Invention

[0019] The present invention provides a mechanism for determining the dimensions of elements within plug-in resources (such as content and/or applications), so as to enable generation of web usage overlays for areas of a web page occupied by such plug-ins.

[0020] According to one aspect of the invention, an element within the plug-in resources is moved to one or more corners of the output region of the plug-in resource. Changes in dimensions of the output region are detected, and the dimensions of the element are calculated. The element is then restored to its original position. The information regarding changes in dimensions is generally available from third-party plug-in applications and can be readily extracted therefrom.

[0021] Once the dimensions of the element are known, an overlay can be generated that visually aligns with the plug-in resource. Thus, when the overlay is presented, color-coding and other visual features are properly positioned and sized within the overall display.

[0022] The present invention can be used for overlay generation for any type of plug-in content, such as for example a movie presentation, whether interactive or non-interactive. One example of such a movie presentation is that provided by an application such as Flash, provided by Macromedia of San Francisco, California. For example, it may be desirable to provide a color-coded overlay indicating related popularity and/or traffic for various interactive components (such as buttons) in a Flash movie. One skilled in the art will recognize that the techniques described herein can also be applied to other types of plug-ins, including for example QuickTime movies, Java applets, and the like.

[0023] The present invention can also be used for generating other types of usage reports, including for example hard copy output, text or graphical reports, and the like.

Brief Description of the Drawings

[0024] The accompanying drawings illustrate several embodiments of the invention and, together with the description, serve to explain the principles of the invention.

[0025] Fig. 1 is a block diagram depicting an architecture for website traffic data collection according to one embodiment of the present invention.

[0026] Fig. 2 is a screen shot depicting an example of a page analysis report according to one embodiment.

[0027] Figs. 3A through 3D depict an example of a method for determining the dimensions of an element of a plug-in resource, according to one embodiment.

[0028] Fig. 4A is a flowchart depicting a method for determining the dimensions of a component of a plug-in resource, where the origin of the element is not necessarily in the center of the element, according to one embodiment.

[0029] Fig. 4B is a flowchart depicting a method for determining the dimensions of a component of a plug-in resource, where the origin of the element is in the center of the element, according to one embodiment.

[0030] Fig. 5 is a block diagram depicting an architecture for modifying a third-party plug-in resource so as to provide usage tracking functionality according to one embodiment.

[0031] Fig. 6 is a flowchart depicting a method for modifying a third-party plug-in resource so as to provide usage tracking functionality according to one embodiment.

[0032] Fig. 7 is a flowchart depicting a method for generating a usage overlay for a plug-in resource, according to one embodiment.

[0033] Fig. 8 is a screen shot depicting an example of an overlay report on a third-party plug-in resource.

[0034] One skilled in the art will recognize that these Figures are merely examples of the operation of the invention according to one embodiment, and that other architectures and modes of operation can be used without departing from the essential characteristics of the invention.

Detailed Description of the Embodiments

[0035] The present invention is now described more fully with reference to the accompanying Figures, in which several embodiments of the invention are shown. The present invention may be embodied in many different forms and should not be construed as limited to the embodiments set forth herein. Rather these embodiments are provided so that this disclosure will be complete and will fully convey the invention to those skilled in the art.

[0036] For illustrative purposes, the invention is described with respect to interactive elements (such as buttons) within a plug-in resource such as content to be displayed within a web page. One example of such a plug-in resource is a content item interpreted and displayed by the Flash plug-in, available from Macromedia, Inc. of San Francisco, California; the Flash plug-in works in conjunction with a conventional web browser such as Internet Explorer, available from Microsoft Corporation of Redmond, Washington. The Flash plug-in provides tools for presentation of video and interactive content within the context of a web page. One skilled in the art will recognize that the present invention can be used for determining dimensions of elements in other contexts as well, and that the present invention can be used in connection with other types of plug-in resources.

Architecture

[0037] Referring now to Fig. 1, there is shown a block diagram depicting a system for website traffic data collection according to one embodiment of the present invention. User 112 interacts with client machine 107, which runs a software application such as browser 110 for accessing and displaying web pages. Client machine 107 may be an ordinary personal computer, including well-known components such as a CPU running an operating system such as Microsoft Windows, a keyboard, mouse, display screen, and Internet connection (not shown). Client machine 107 may run various software applications in addition to browser 110. Browser 110 includes scripting engine 116, such as JavaScript, as is commonly found in commercially available browsers. In response to a user 112 action such as clicking on a link or typing in a URL, client machine 107 issues a web page request 111 that is transmitted via the Internet to content server 101. In response to request 111, content server 101 transmits web page 102 (in the form of HTML code, for example) to client machine 107. Browser 110 displays the requested web page 102 on client machine 107.

[0038] Web page 102 includes beacon code, which in one embodiment is a pointer to a beacon (such as a 1 pixel by 1 pixel transparent image). The beacon is typically invisible to the user, such as a transparent one-pixel image. For purposes of the following description, a beacon is any element that is embedded in a web page 102 which is loaded automatically by browser 110 that references an external server 106 and is used to monitor traffic. The beacon code can be provided as a script (such as a JavaScript script) to be executed by scripting engine 116. The beacon code causes client machine 107 to generate resource requests 105 to data collection server 106. These resource requests 105 are usually dynamically generated according to the script instructions. Data collection server 106 records such requests in a log 108, and can also record additional information associated with the request (such as the date and time, and possibly some identifying information that may be encoded in the resource request). Thus, tracking server 106 records the occurrence of a "hit" to web page 102. Tracking server 106 also transmits the requested one-pixel image to client machine 107 so that the resource request is satisfied.

[0039] Analysis module 113 retrieves stored tracking data from log 108, filters the data, and outputs reports 114 to a web administrator 115. Reports 114 may be provided in hard copy, or via a display screen (not shown), or by some other means. Reports 114 include, for example, overviews and statistical analyses describing the relative frequency with which

various site paths are being followed through the website. Examples of such reports are described below.

[0040] Module 113 may be implemented in software running on server 106 or on another computer that can access log 108.

[0041] In one embodiment, communications between client machine 107, content server 101, and data collection server 106 are accomplished using well known network protocols, such as TCP/IP and HTTP, for communication across the Internet. Other communication methodologies and protocols can also be used.

[0042] For example, referring now to Fig. 2, there is shown an example of a page analysis report 201, displayed alongside an image of the web page 102 being analyzed. In one embodiment, report 201 is provided to a site administrator or owner interacting with data collection server 106.

[0043] In the example of Fig. 2, report 201 includes identification 202 of the website and web page being analyzed; report date 203; report options and settings 204; page metrics 205; and links 206 to related reports. In addition, variable levels and shades of color density are superimposed on the displayed view of web page 102, in order to visually represent the relative number of clicks each item 208 or screen region has received. Color key 207 is a legend to indicate the meaning of various superimposed colors. According to the techniques described herein, an overlay report such as shown in Fig. 2 can be generated for areas of the screen occupied by plug-in content, including third-party plug-in content.

[0044] Fig. 8 shows an example of an overlay report 800 for a web page that includes third-party plug-in content such as areas 801. In this example, areas 801 represent Flash content that occupies a portion of the web page. Certain portions 802 of content areas 801 are highlighted in different colors to denote the traffic levels. Additional information regarding web traffic is shown in overlapping windows 803.

[0045] In order to track user interaction with plug-in content, tracking code is integrated into the plug-in code. Referring now to Fig. 6, there is shown a flowchart depicting a method for modifying a third-party plug-in resource so as to provide usage tracking functionality according to one embodiment. Referring also to Fig. 5, there is shown a block diagram depicting an architecture for modifying a third-party plug-in resource so as to provide usage track-

ing functionality according to one embodiment. Plug-in application 501 represents an application, such as Flash, for displaying and presenting plug-in content.

[0046] Client machine 107 sends a web page request 111 to content server 101. In response, content server 101 provides plug-in content 502 to client machine 107. In one embodiment, prior to providing content 502, content server 101 inserts tracking code. This tracking code includes code for identifying and reporting on user interaction with various elements of the plug-in resource. The tracking code replaces and/or supplements at least one method within the received plug-in content, so as to enable transmission of user behavior to a tracking component.

[0047] Browser 110 runs 604 the modified plug-in content 502. Browser 110 displays web page output 506 including plug-in output. User actions with respect to tracked elements (such as a user clicking on a button or otherwise interacting with an element) are detected 605. Usage/behavior data 507 is sent 606 to data collection server 106 according to techniques described herein; the data is stored in log 108. Analysis module 113 uses the collected data is then used for generating reports 607; in one embodiment, analysis module 113 includes web overlay generation module 507 for generating overlay reports 508, which are then output. Reports 607 can thus indicate patterns of aggregate user behavior with respect to plug-in content.

[0048] Referring now to Fig. 7, there is shown a method for generating a usage overlay for a plug-in resource containing interactive elements, according to one embodiment. In one embodiment, analysis module 113 as shown in Fig. 1 performs the steps of Fig. 7; in other embodiments, these steps can be performed by any component of a web analysis tracking system.

[0049] First, the locations and dimensions of elements are determined 701. In one embodiment, the locations (in terms of x,y coordinates) of elements are readily available, and the dimensions are determined using techniques described in more detail herein. Data describing user interaction with these elements is received 702; in one embodiment, this step involves retrieved data from log 108 containing records of user interactions. In one embodiment, this data is aggregated so that the generated report indicates overall behavior trends for a large group of users. Based on the determined locations and dimensions of elements, and based on the aggregated data describing user behavior, an overlay is generated 703. In one

embodiment the overlay is a visual representation of relative levels of activity for various areas within the plug-in resource; for example, high-activity areas can be shown in one color, and lower-activity areas can be shown in a different color. Since the component generating the overlay is aware of the positions and dimensions of the elements, the positions and dimensions colored areas of the overlay can be made to match those of the elements to which they refer. The resulting report, output in step 704, shows a representation of the plug-in content, with a color-coded overlay indicating relative activity for various elements within the plug-in content.

[0050] Referring now to Figs. 4A and 4B, there are shown two methods for determining the dimensions of an element of a plug-in resource such as a component of a Flash movie. Fig. 4A depicts a method where the origin of the element is not necessarily in the center of the element (it may be at a corner, or at some other location). Referring also to Figs. 3A through 3D, there is shown an example of the method. Fig. 4B depicts a method where the origin of the element is known to be in the center of the element.

[0051] Although in Figs. 3A through 3D, the element is shown at various positions within the output region, the steps herein are performed without displaying these moved elements to the user; thus the method is done transparently without redrawing the plug-in resource. Output to the user is therefore not disrupted by the use of the method described herein.

[0052] Fig. 3A shows element 302 in its original position, with an origin point 303. X_c and Y_c are the unknown dimensions of element 302. Output region 301 represents an area, for example within a web page, for the display of web plug-in content.

[0053] Referring now to Fig. 4A, first the overall dimensions X_{orig},Y_{orig} of output region 301 are noted 409. Then, element 302 is moved 401 so that its origin point 303 is at the top left corner of output region 301 (Fig. 3B). In one embodiment, existing methods within the plug-in application are used to move element 302; for example, the Macromedia Flash plug-in contains methods that enable movement of elements and other objects.

[0054] Many plug-in applications automatically resize their output regions to account for elements and objects that have moved beyond the original region. For example, the Macromedia Flash plug-in automatically resizes the Flash content viewing area (output region) if it detects that an element has moved outside the viewing area.

[0055] Accordingly, since origin 303 is located within element 302, movement of origin 303 to a corner causes some portion of element 302 to be located outside the original output region 301. Once the plug-in code has automatically resized output region 301, as shown in Fig. 3B, the new dimensions of the overall Flash movie 301 are compared to the original dimensions X_{orig}, Y_{orig} 402. The differential in both the X and Y dimensions is stored as X_1, Y_1; this accounts for a first portion of the dimensions of element 302.

[0056] Next, element 302 is moved 404 so that its origin 303 is at the bottom right corner of output region 301 (Fig. 3C). Again, the plug-in code automatically changes the dimensions of output region 301; this difference in dimensions is noted 405 as X_2, Y_2. This increase in X and Y dimensions provides the other portion of the X and Y coordinates. The original size of output region 301 is restored after the increased dimensions are determined. Element 302 is then moved back 406 to its original location (Fig. 3D). Element 302 dimensions are then calculated 407 as $X_c = X_1 + X_2$ and $Y_c = Y_1 + Y_2$.

[0057] Once element 302 dimensions are known, an overlay can be generated according to the techniques described above, and the overlay will have the correct location and the correct dimensions.

[0058] Referring now to Fig. 4B, there is shown an embodiment where origin point 303 is known to be located at the center of element 302. Here, steps 409, 401, and 402 are performed as above. However, element 302 need not be moved to a second corner; rather it is returned 406 to its original location. The values X_1 and Y_1, are each doubled 420 to provide the total X and Y dimensions of element 302. Thus $X_c = 2*X_1$ and $Y_c = 2* Y_1$.

[0059] One skilled in the art will recognize that the above-described technique can be applied in any situation where it is desirable to display an overlay over a portion of a web page occupied by plug-in content. This can apply, for example, to Java applets, Windows Media Player items, Quicktime movies, or the like.

[0060] In the above description, for purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of the invention. It will be apparent, however, to one skilled in the art that the invention can be practiced without these specific details. In other instances, structures and devices are shown in block diagram form in order to avoid obscuring the invention.

[0061] In particular, one skilled in the art will recognize that other architectures and analysis and processing techniques and mechanisms may be used, and that the present invention can be implemented using mechanisms other than those described above.

[0062] Reference in the specification to "one embodiment" or "an embodiment" means that a particular feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment of the invention. The appearances of the phrase "in one embodiment" in various places in the specification are not necessarily all referring to the same embodiment.

[0063] Some portions of the detailed description are presented in terms of algorithms and symbolic representations of operations on data bits within a computer memory. These algorithmic descriptions and representations are the means used by those skilled in the data processing arts to most effectively convey the substance of their work to others skilled in the art. An algorithm is here, and generally, conceived to be a self-consistent sequence of steps leading to a desired result. The steps are those requiring physical manipulations of physical quantities. Usually, though not necessarily, these quantities take the form of electrical or magnetic signals capable of being stored, transferred, combined, compared, and otherwise manipulated. It has proven convenient at times, principally for reasons of common usage, to refer to these signals as bits, values, elements, symbols, characters, terms, numbers, or the like.

[0064] It should be borne in mind, however, that all of these and similar terms are to be associated with the appropriate physical quantities and are merely convenient labels applied to these quantities. Unless specifically stated otherwise as apparent from the discussion, it is appreciated that throughout the description, discussions utilizing terms such as "processing" or "computing" or "calculating" or "determining" or "displaying" or the like, refer to the action and processes of a computer system, or similar electronic computing device, that manipulates and transforms data represented as physical (electronic) quantities within the computer system's registers and memories into other data similarly represented as physical quantities within the computer system memories or registers or other such information storage, transmission or display devices.

[0065] The present invention also relates to an apparatus for performing the operations herein. This apparatus may be specially constructed for the required purposes, or it may

comprise a general-purpose computer selectively activated or reconfigured by a computer program stored in the computer. Such a computer program may be stored in a computer readable storage medium, such as, but is not limited to, any type of disk including floppy disks, optical disks, CD-ROMs, and magnetic-optical disks, read-only memories (ROMs), random access memories (RAMs), EPROMs, EEPROMs, magnetic or optical cards, or any type of media suitable for storing electronic instructions, and each coupled to a computer system bus.

[0066] The algorithms and modules presented herein are not inherently related to any particular computer or other apparatus. Various general-purpose systems may be used with programs in accordance with the teachings herein, or it may prove convenient to construct more specialized apparatuses to perform the method steps. The required structure for a variety of these systems will appear from the description below. In addition, the present invention is not described with reference to any particular programming language. It will be appreciated that a variety of programming languages may be used to implement the teachings of the invention as described herein. Furthermore, as will be apparent to one of ordinary skill in the relevant art, the modules, features, attributes, methodologies, and other aspects of the invention can be implemented as software, hardware, firmware or any combination of the three. Of course, wherever a component of the present invention is implemented as software, the component can be implemented as a standalone program, as part of a larger program, as a plurality of separate programs, as a statically or dynamically linked library, as a kernel loadable module, as a device driver, and/or in every and any other way known now or in the future to those of skill in the art of computer programming. Additionally, the present invention is in no way limited to implementation in any specific operating system or environment.

[0067] It will be understood by those skilled in the relevant art that the above-described implementations are merely exemplary, and many changes can be made without departing from the true spirit and scope of the present invention. Therefore, it is intended by the appended claims to cover all such changes and modifications that come within the true spirit and scope of this invention.

Claims

What is claimed is:

1. A method for determining dimensions of an element within a plug-in resource having a rectangular output region, the element having an origin point, the method comprising:

moving the element so that its origin point is at a first corner of the output region;

measuring a first increase in a horizontal dimension of the output region resulting from moving the resource;

measuring a first increase in a vertical dimension of the output region resulting from moving the resource;

determining, from the measured increases in output region dimensions, the dimensions of the element; and

restoring the element to its original position within the output region.

2. The method of claim 1, wherein the element comprises a button.

3. The method of claim 1, wherein the element comprises a user-activatable region within the output region.

4. The method of claim 1, wherein the origin point is located at the center of the element, and wherein determining the dimensions of the element comprises:

determining a horizontal dimension of the element as equal to twice the measured first increase in the horizontal dimension of the output region; and

determining a vertical dimension of the element as equal to twice the measured first increase in the vertical dimension of the output region.

5. The method of claim 1, wherein the origin point is not located at the center of the element.

6. The method of claim 1, further comprising, prior to determining the dimensions of the element:

moving the element so that its origin point is at a second corner of the output region, the second corner being diagonally opposite the first corner;

measuring a second increase in a horizontal dimension of the output region resulting from moving the resource;

measuring a second increase in a vertical dimension of the output region resulting from moving the resource;

and wherein determining the dimensions of the element comprises:

determining a horizontal dimension of the element as equal to the sum of the first and second measured increases in the horizontal dimension of the output region; and

determining a vertical dimension of the element as equal to the sum of the first and second measured increases in the vertical dimension of the output region.

7. The method of claim 1, wherein the plug-in resource comprises an application and the rectangular output region comprises an area within a web page.

8. The method of claim 1, wherein the plug-in resource comprises a content item for an application and the rectangular output region comprises an area within a web page.

9. The method of claim 1, wherein the plug-in resource comprises a Flash item.

10. The method of claim 1, wherein the plug-in resource comprises video content.

11. The method of claim 1, wherein moving, measuring, determining, and restoring steps are performed responsive to receiving user input activating the element.

12. The method of claim 1, further comprising outputting the determined dimensions of the element.

13. The method of claim 1, further comprising generating and outputting a usage display using the determined dimensions of the element.

14. The method of claim 1, further comprising sending a message including the determined dimensions of the element to a software component.

15. A method of generating a usage report for a plug-in resource on a web page, comprising:

receiving a communication indicating user activation of an object within a plug-in resource, the communication comprising an object identifier and additional identifying indicia for the object;

comparing the object identifier and the additional identifying indicia of the user-activated object with records comprising object identifiers and additional identifying indicia;

responsive to the object identifier of the user-activated object being within a predetermined tolerance factor with respect to an object identifier of a stored record, and the identifier of the stored record having additional identifying indicia matching the additional identifying indicia of the user-activated object:

indicating a match between the user-activated object and the stored record; and

generating a report including a representation of the web page including the plug-in resource by superimposing visual indicators quantifying user activations for objects on the representation of the web page including the plug-in resource.

16. The method of claim 15, wherein the visual indicators are color-coded.

17. A method of generating a usage report for a plug-in resource on a web page, the plug-in resource comprising an object having an object identifier and additional identifying indicia, comprising:

comparing the object identifier and the additional identifying indicia of the object with records comprising object identifiers and additional identifying indicia;

responsive to the object identifier of the object being within a predetermined tolerance factor with respect to an object identifier of a stored record, and the identifier of the stored record having additional identifying indicia matching the additional identifying indicia of the object:

indicating a match between the object and the stored record;

generating a report including a representation of the web page including the plug-in resource by superimposing visual indicators quantifying user activations for objects on the representation of the web page including the plug-in resource.

18. The method of claim 17, wherein the visual indicators are color-coded.

19. A report displaying usage of a plug-in resource on a web page, comprising:

a representation of a plug-in resource including objects; and

superimposed on the representation of the plug-in resource, visual indicators quantifying user activations of the objects.

20. The report of claim 19, wherein the visual indicators are color-coded.

21. A method for displaying a report depicting usage of a plug-in resource on a web page, comprising:

generating a representation of a plug-in resource including objects; and

superimposing on the representation of the plug-in resource, visual indicators quantifying user activations of the objects.

22. The method of claim 21, wherein the visual indicators are color-coded.

23. A computer program product for determining dimensions of an element within a plug-in resource having a rectangular output region, the element having an origin point, the computer program product comprising:

a computer-readable medium; and

computer program code, encoded on the medium, for:

moving the element so that its origin point is at a first corner of the output region;

measuring a first increase in a horizontal dimension of the output region resulting from moving the resource;

measuring a first increase in a vertical dimension of the output region resulting from moving the resource;

determining, from the measured increases in output region dimensions, the dimensions of the element; and

restoring the element to its original position within the output region.

24. The computer program product of claim 23, wherein the element comprises a button.

25. The computer program product of claim 23, wherein the element comprises a user-activatable region within the output region.

26. The computer program product of claim 23, wherein the origin point is located at the center of the element, and wherein the computer program code for determining the dimensions of the element comprises computer program code for:

determining a horizontal dimension of the element as equal to twice the measured first increase in the horizontal dimension of the output region; and

determining a vertical dimension of the element as equal to twice the measured first increase in the vertical dimension of the output region.

27. The computer program product of claim 23, wherein the origin point is not located at the center of the element.

28. The computer program product of claim 23, further comprising computer program code for, prior to determining the dimensions of the element:

moving the element so that its origin point is at a second corner of the output region, the second corner being diagonally opposite the first corner;

measuring a second increase in a horizontal dimension of the output region resulting from moving the resource;

measuring a second increase in a vertical dimension of the output region resulting from moving the resource;

and wherein the computer program code for determining the dimensions of the element comprises computer program code for:

determining a horizontal dimension of the element as equal to the sum of the first and second measured increases in the horizontal dimension of the output region; and

determining a vertical dimension of the element as equal to the sum of the first and second measured increases in the vertical dimension of the output region.

29. The computer program product of claim 23, wherein the plug-in resource comprises an application and the rectangular output region comprises an area within a web page.

30. The computer program product of claim 23, wherein the plug-in resource comprises a content item for an application and the rectangular output region comprises an area within a web page.

31. The computer program product of claim 23, wherein the plug-in resource comprises a Flash item.

32. The computer program product of claim 23, wherein the plug-in resource comprises video content.

33. The computer program product of claim 23, wherein the computer program code for moving, measuring, determining, and restoring operate responsive to receiving user input activating the element.

34. The computer program product of claim 23, further comprising computer program code for outputting the determined dimensions of the element.

35. The computer program product of claim 23, further comprising computer program code for generating and outputting a usage display using the determined dimensions of the element.

36. The computer program product of claim 23, further comprising computer program code for sending a message including the determined dimensions of the element to a software component.

37. A computer program product of generating a usage report for a plug-in resource on a web page, comprising:

a computer-readable medium; and

computer program code, encoded on the medium, for:

receiving a communication indicating user activation of an object within a plug-in resource, the communication comprising an object identifier and additional identifying indicia for the object;

comparing the object identifier and the additional identifying indicia of the user-activated object with records comprising object identifiers and additional identifying indicia;

responsive to the object identifier of the user-activated object being within a predetermined tolerance factor with respect to an object identifier of a stored record, and the identifier of the stored record having additional identifying indicia matching the additional identifying indicia of the user-activated object:

indicating a match between the user-activated object and the stored record; and

generating a report including a representation of the web page including the plug-in resource by superimposing visual indicators quantifying

user activations for objects on the representation of the web page including the plug-in resource.

38. The computer program product of claim 37, wherein the visual indicators are color-coded.

39. A computer program product of generating a usage report for a plug-in resource on a web page, the plug-in resource comprising an object having an object identifier and additional identifying indicia, comprising:

a computer-readable medium; and

computer program code, encoded on the medium, for:

comparing the object identifier and the additional identifying indicia of the object with records comprising object identifiers and additional identifying indicia;

responsive to the object identifier of the object being within a predetermined tolerance factor with respect to an object identifier of a stored record, and the identifier of the stored record having additional identifying indicia matching the additional identifying indicia of the object:

indicating a match between the object and the stored record;

generating a report including a representation of the web page including the plug-in resource by superimposing visual indicators quantifying user activations for objects on the representation of the web page including the plug-in resource.

40. The computer program product of claim 39, wherein the visual indicators are color-coded.

41. A computer program product for displaying a report depicting usage of a plug-in resource on a web page, comprising:

a computer-readable medium; and

computer program code, encoded on the medium, for:

generating a representation of a plug-in resource including objects; and

superimposing on the representation of the plug-in resource, visual indicators quantifying user activations of the objects.

42. The computer program product of claim 41, wherein the visual indicators are color-coded.

43. A system for determining dimensions of an element within a plug-in resource having a rectangular output region, the element having an origin point, the system comprising:

a content server, for receiving a web page request and for providing, to a client machine, modified plug-in content, the modified plug-in content comprising code for determining dimensions of an element within a plug-in resource by:

moving the element so that its origin point is at a first corner of the output region;

measuring a first increase in a horizontal dimension of the output region resulting from moving the resource;

measuring a first increase in a vertical dimension of the output region resulting from moving the resource;

determining, from the measured increases in output region dimensions, the dimensions of the element; and

restoring the element to its original position within the output region;

a data collection server for receiving usage data with respect to the plug-in content;

an analysis module for analyzing the received usage data; and

a report generator for generating at least one overlay report based on the analyzed usage data.

44. The system of claim 43, wherein the element comprises a button.

45. The system of claim 43, wherein the element comprises a user-activatable region within the output region.

46. The system of claim 43, wherein the plug-in resource comprises an application and the rectangular output region comprises an area within a web page.

47. The system of claim 43, wherein the plug-in resource comprises a content item for an application and the rectangular output region comprises an area within a web page.

48. The system of claim 43, wherein the plug-in resource comprises a Flash item.

49. The system of claim 43, wherein the plug-in resource comprises video content.

50. The system of claim 43, wherein the code causes the moving, measuring, determining, and restoring steps to be performed responsive to receiving user input activating the element.

51. The system of claim 43, wherein the overlay report comprises a usage display using the determined dimensions of the element.

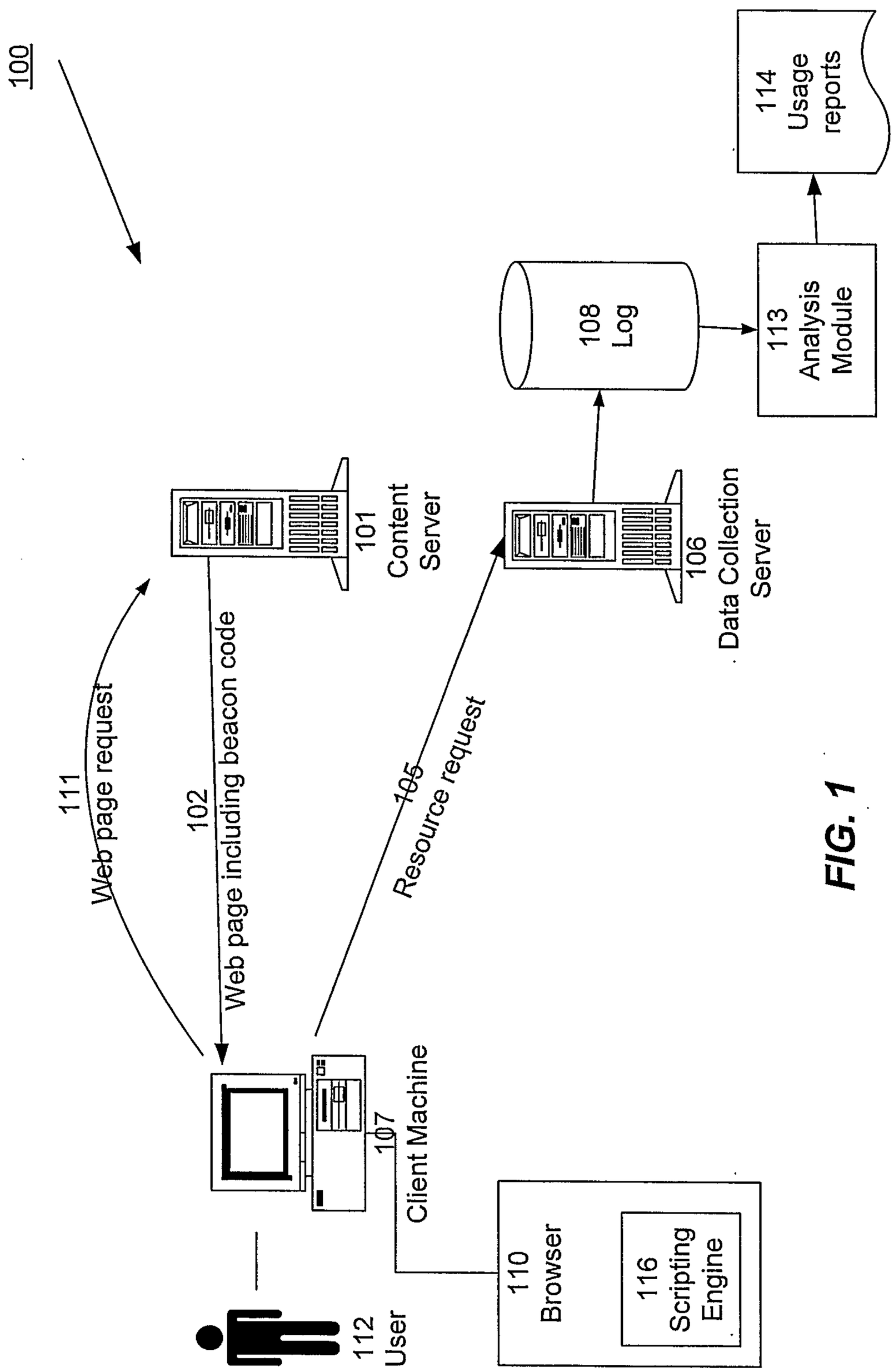

100
114
Usage
reports
108
Log
113
Analysis
Module
101
Content
Server
106
Data Collection
Server
111
Web page request
102
Web page including beacon code
105
Resource request
107
Client Machine
112
User
110
Browser
116
Scripting
Engine

FIG. 1

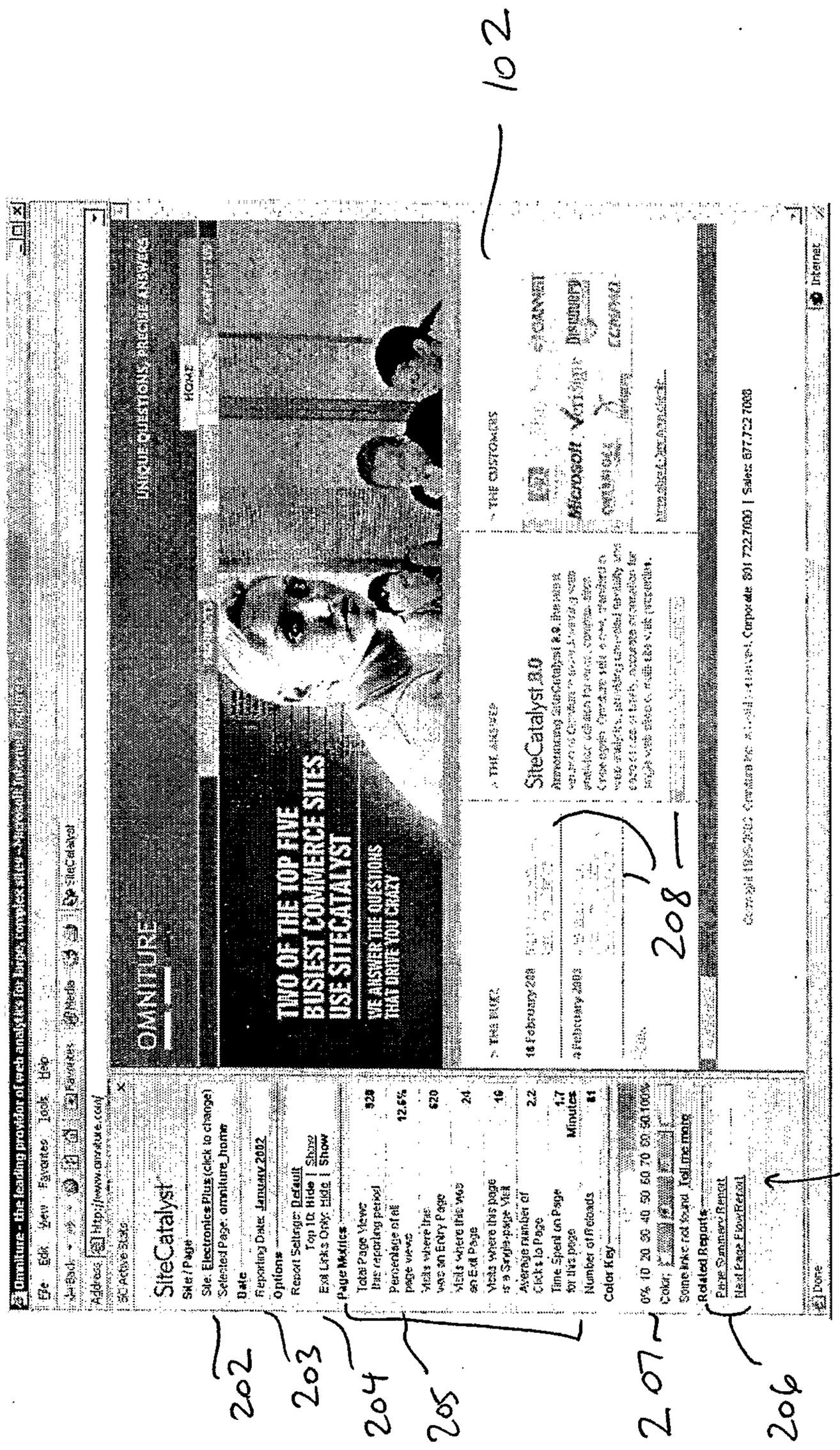

102
UNIQUE QUESTIONS. PRECISE ANSWERS.
HOME
OMNITURE
TWO OF THE TOP FIVE BUSIEST COMMERCE SITES USE SITECATALYST
WE ANSWER THE QUESTIONS THAT DRIVE YOU CRAZY
SiteCatalyst
Site: Electronics Plus (click to change)
Selected Page: omniture_home
Date
Reporting Date: January 2002
Options
Reset Settings: Default
Top 10: Hide | Show
Exit Links Only: Hide | Show
Page Metrics
Total Page Views this reporting period 928
Percentage of all page views 12.5%
Visits where this was an Entry Page 620
Visits where this was an Exit Page 24
Visits where this page is a Single-page Visit 16
Average number of Clicks to Page 2.2
Time Spent on Page for this page 1:7 Minutes
Number of Refreads 81
Color Key
0% 10 20 30 40 50 60 70 80 90 100%
Color:
Some links not found. Tell me more
Related Reports
Page Summary Report
Next Page Flow Report
THE CUSTOMERS
THE BUYER
THE ANSWER
SiteCatalyst 3.0
16 February 2003
8 February 2003
Done
Internet
201
202
203
204
205
207
208
206

FIG. 2

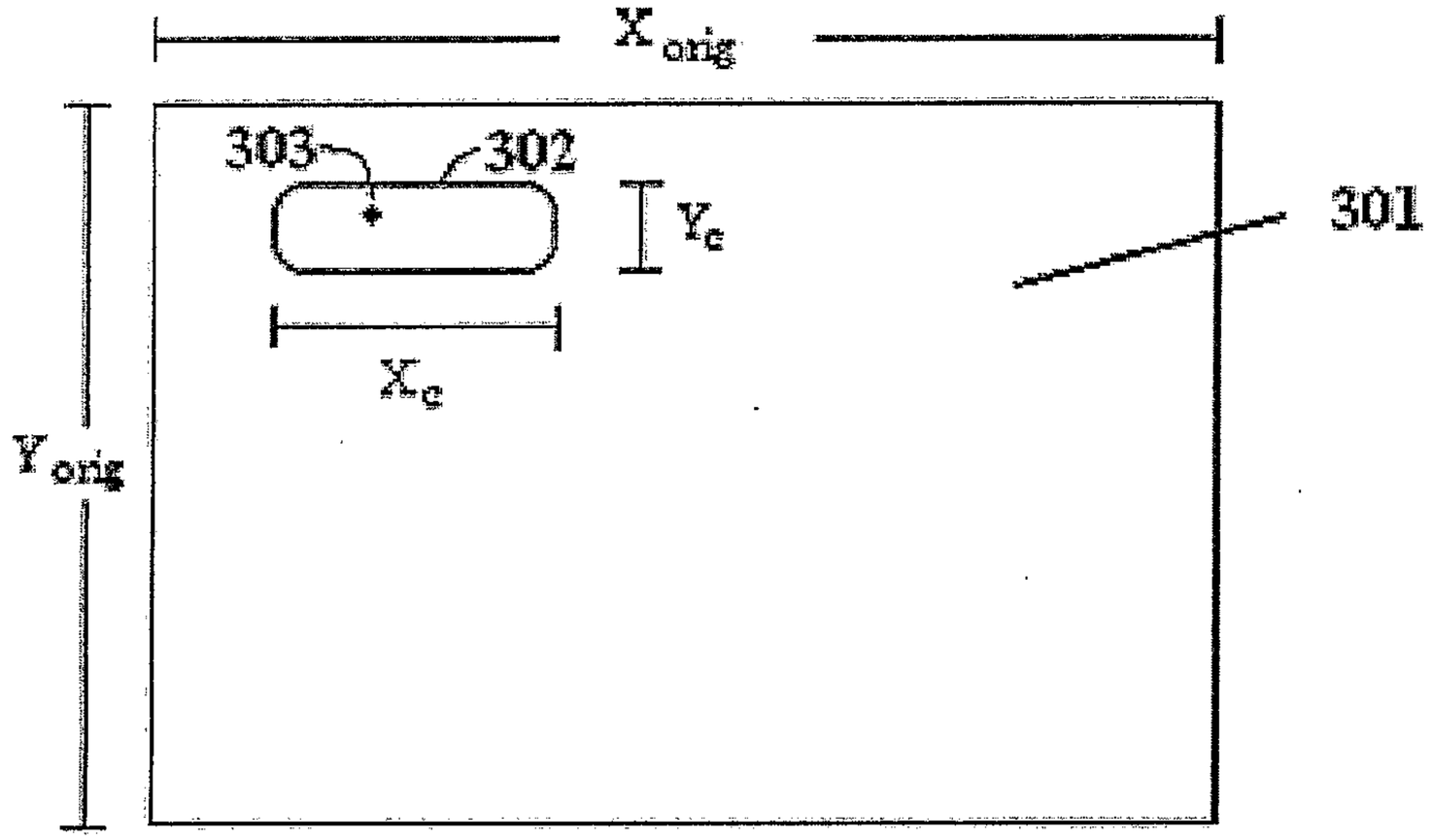


Fig. 3A

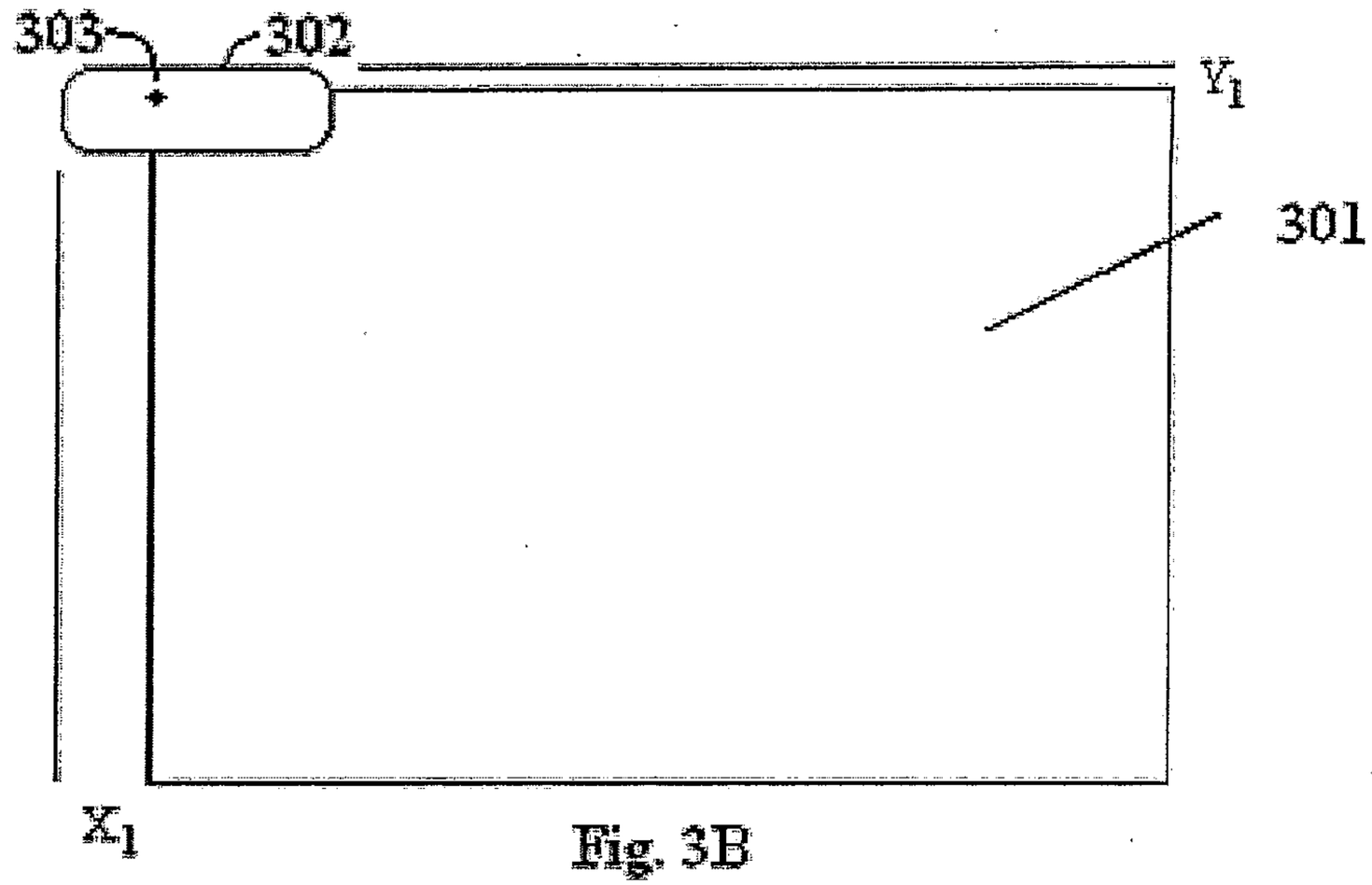


Fig. 3B

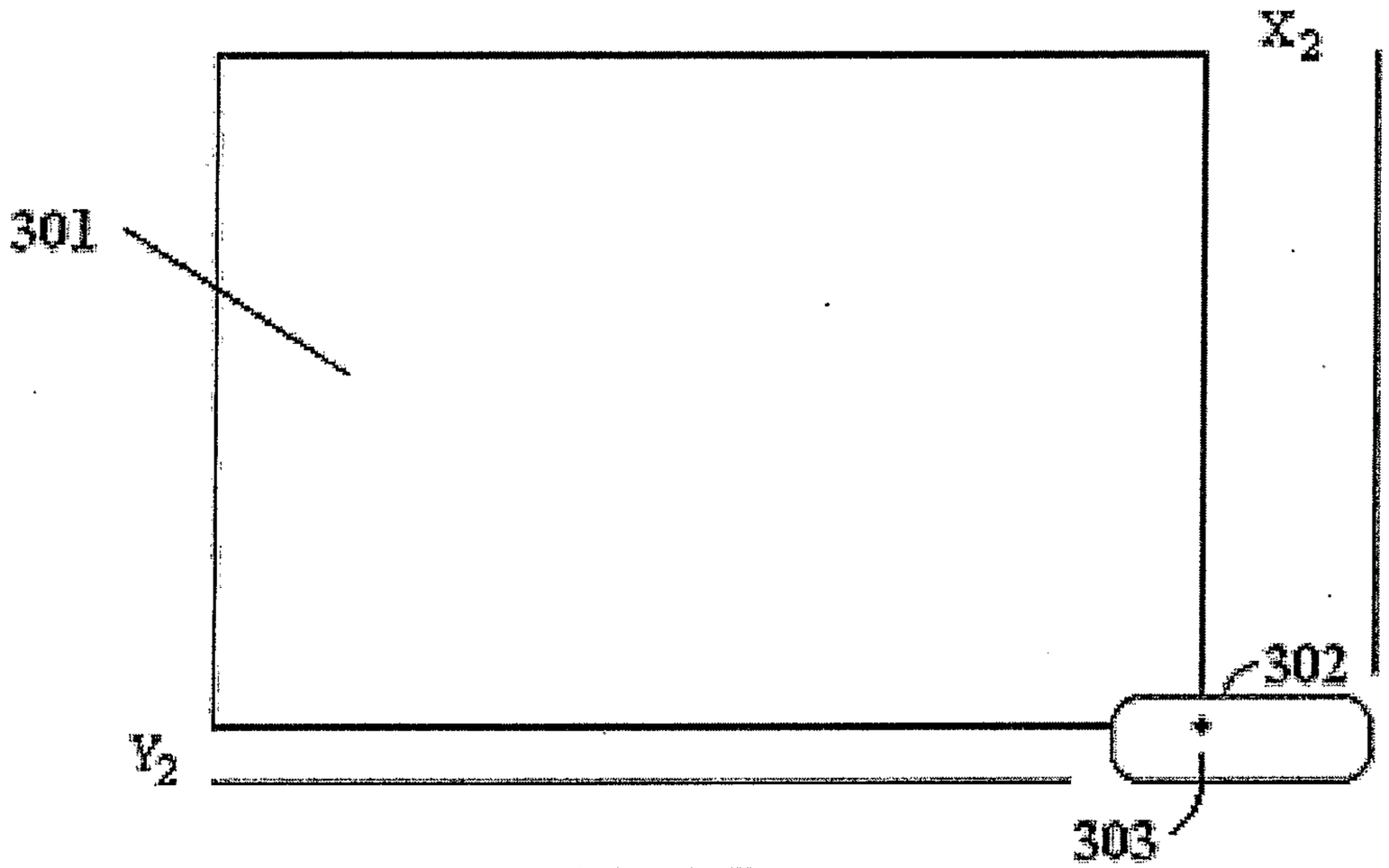


Fig. 3C

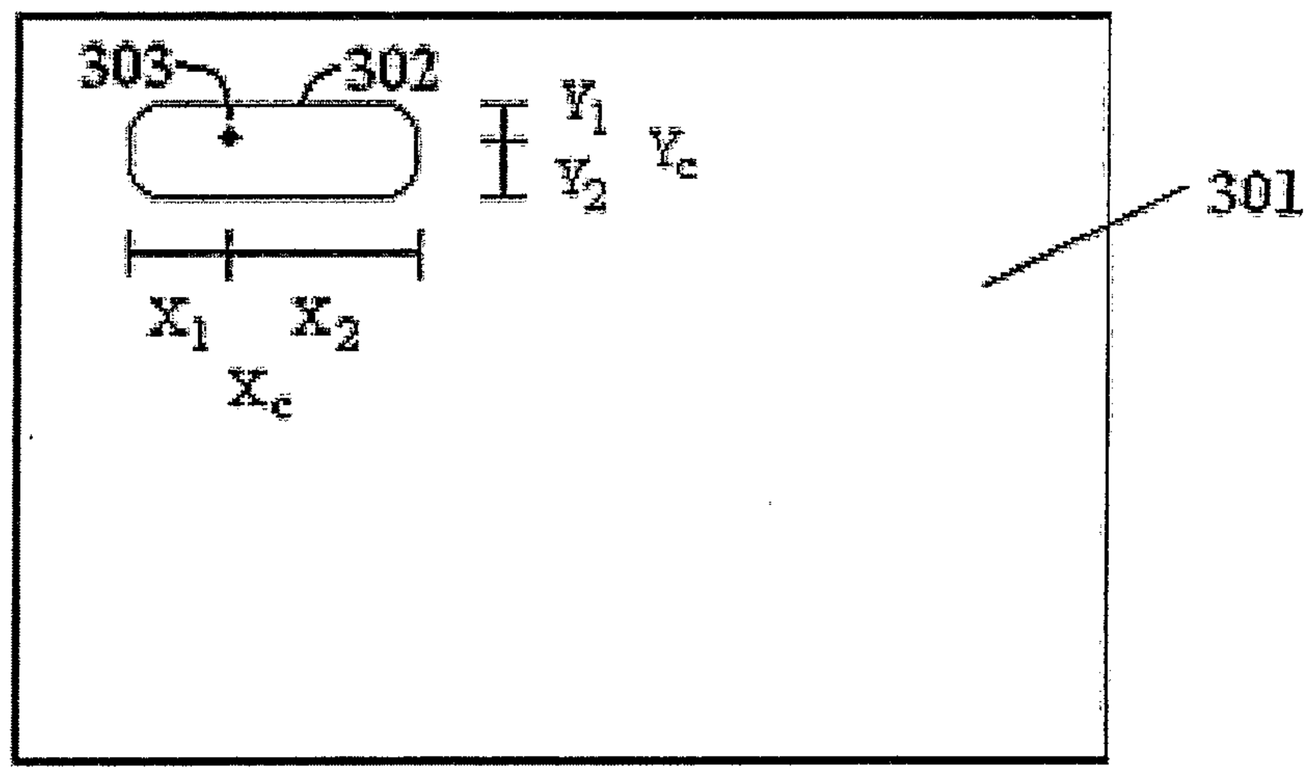


Fig. 3D

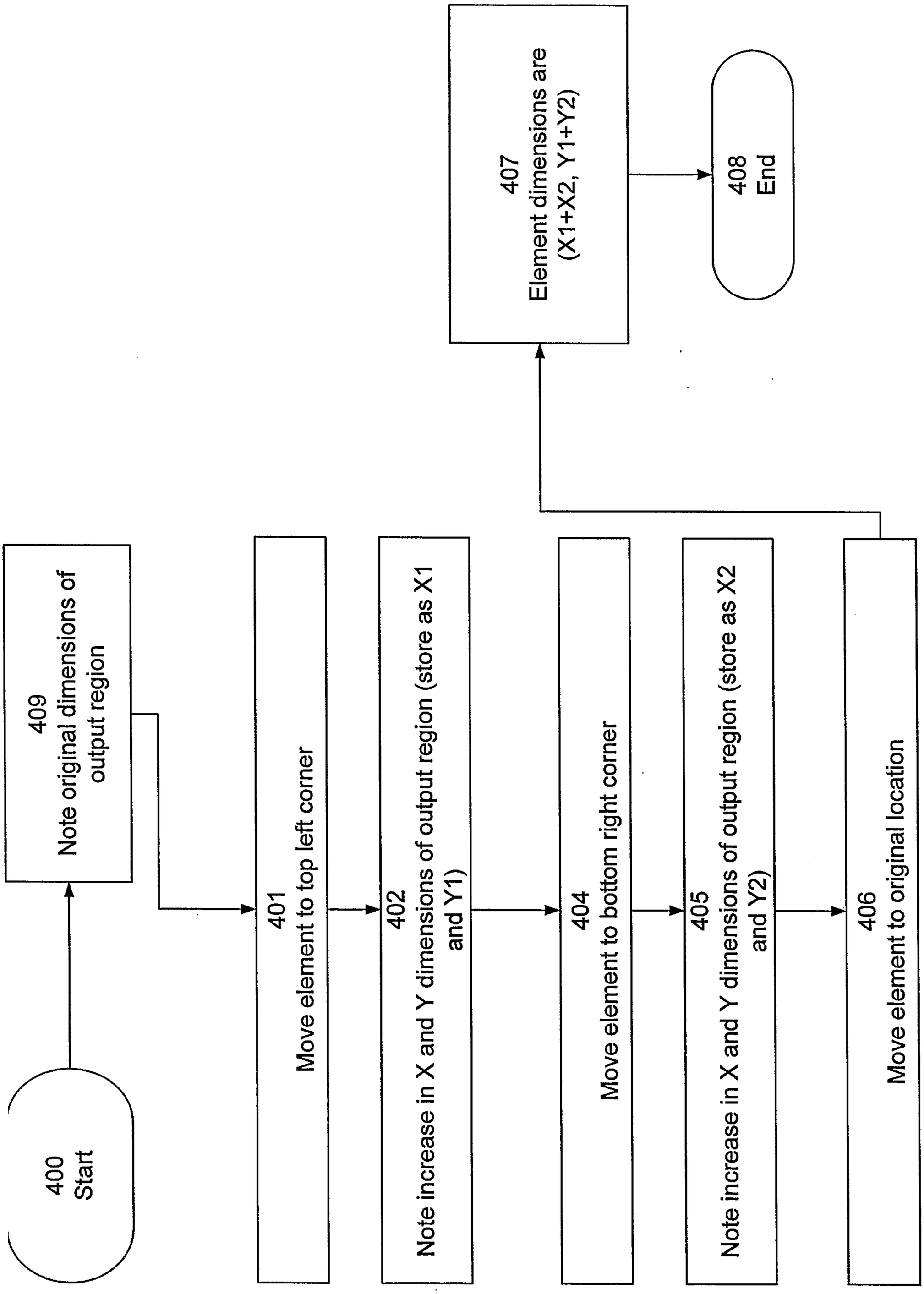

400
Start
409
Note original dimensions of output region
401
Move element to top left corner
Note increase in X and Y dimensions of output region (store as X1 and Y1)
402
404
Move element to bottom right corner
Note increase in X and Y dimensions of output region (store as X2 and Y2)
405
406
Move element to original location
407
Element dimensions are (X1+X2, Y1+Y2)
408
End

FIG. 4A

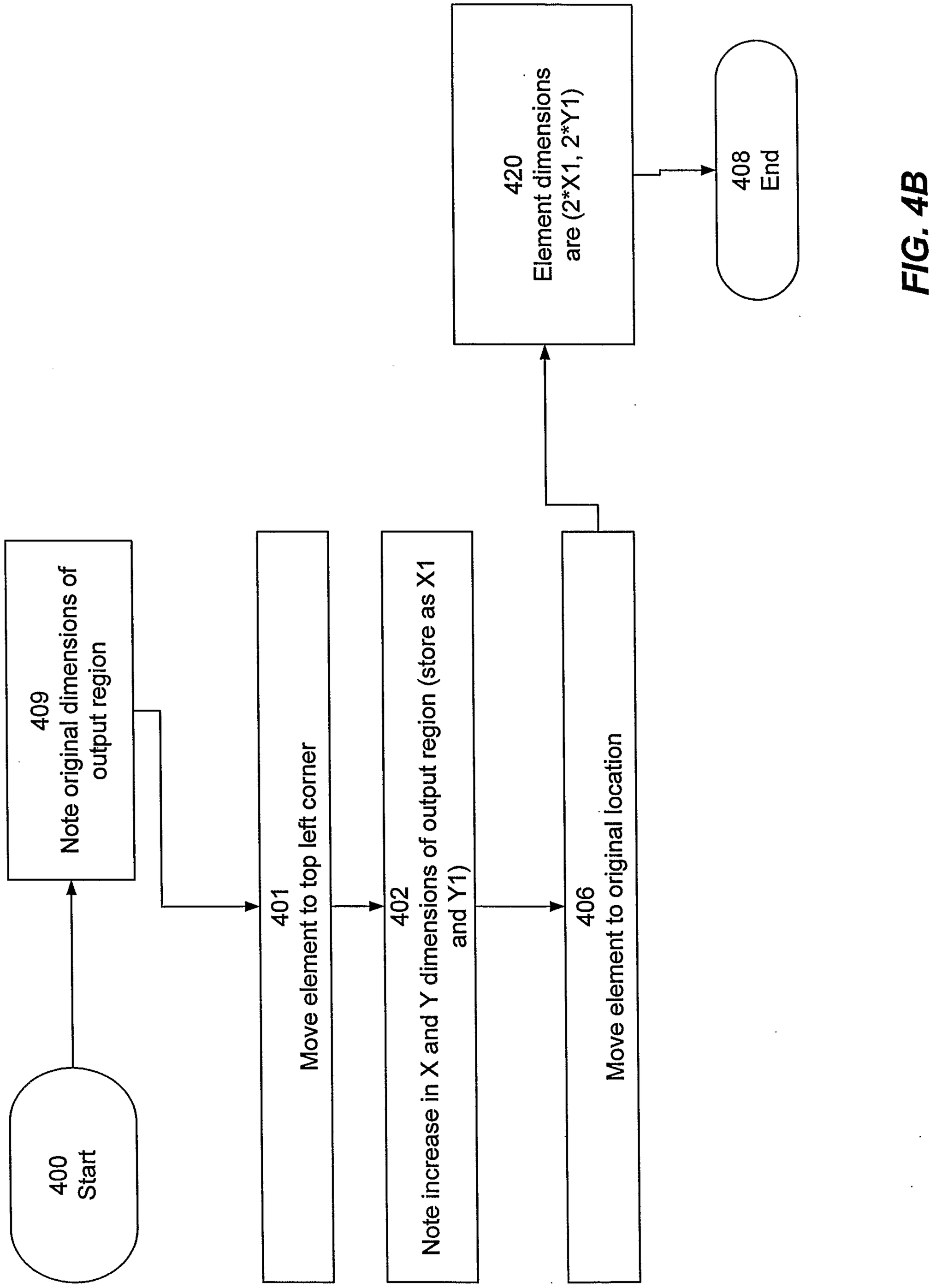

400
Start
409
Note original dimensions of output region
401
Move element to top left corner
402
Note increase in X and Y dimensions of output region (store as X1 and Y1)
406
Move element to original location
420
Element dimensions are (2*X1, 2*Y1)
408
End

FIG. 4B

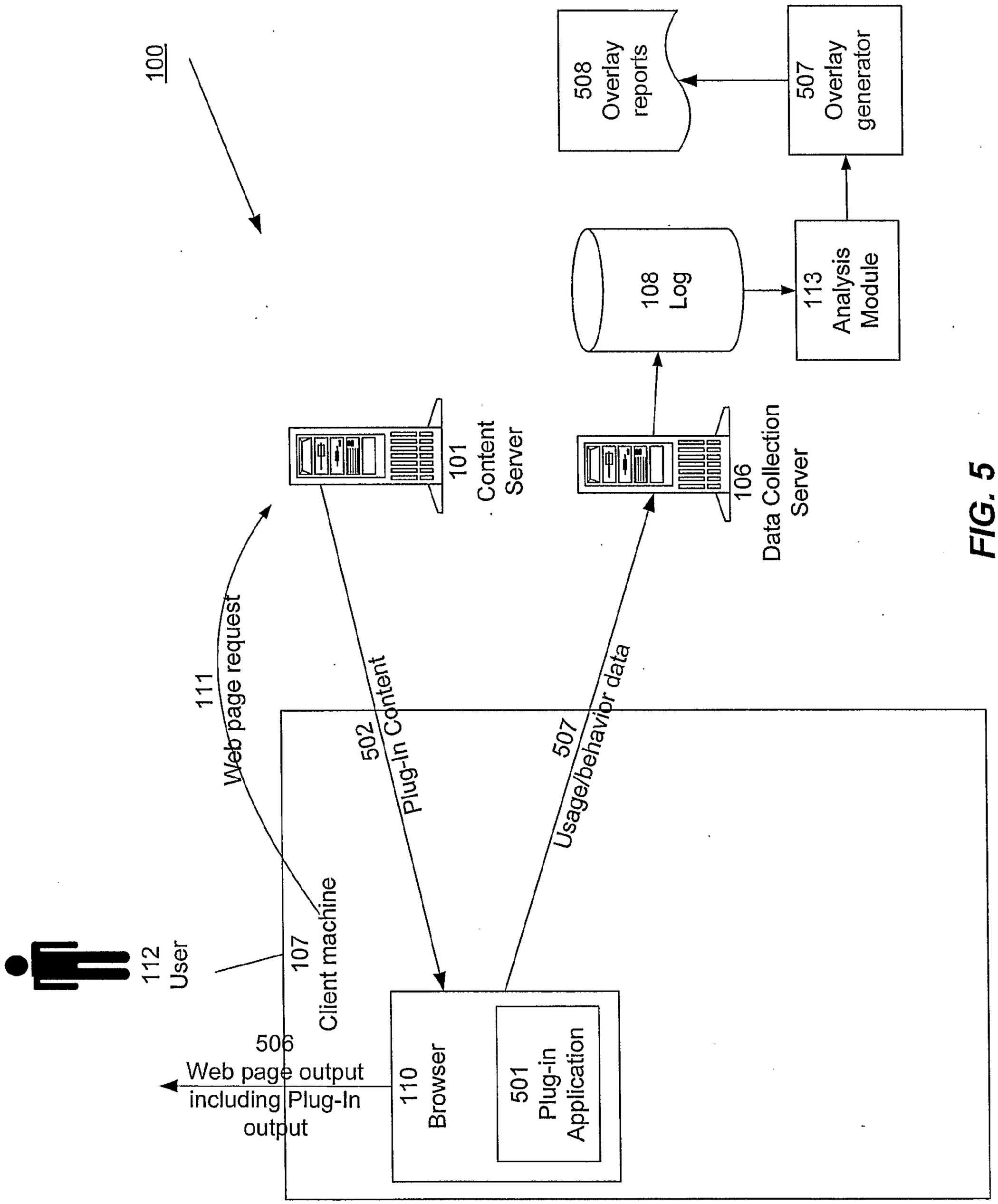

100
508 Overlay reports
507 Overlay generator
108 Log
113 Analysis Module
101 Content Server
106 Data Collection Server
111 Web page request
502 Plug-In Content
507 Usage/behavior data
112 User
107 Client machine
506 Web page output including Plug-In output
110 Browser
501 Plug-in Application

FIG. 5

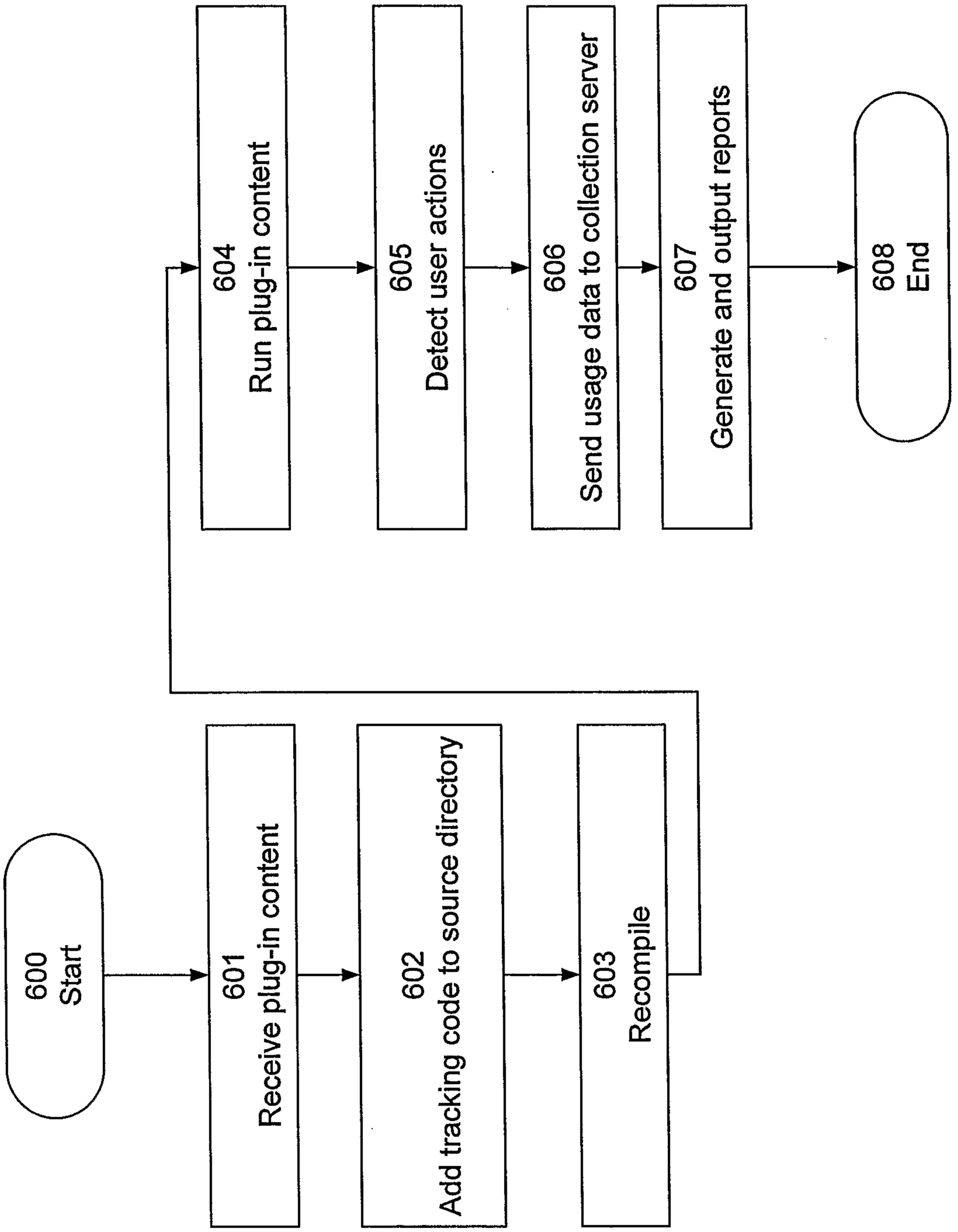

600
Start
601
Receive plug-in content
602
Add tracking code to source directory
603
Recompile
604
Run plug-in content
605
Detect user actions
606
Send usage data to collection server
607
Generate and output reports
608
End

FIG. 6

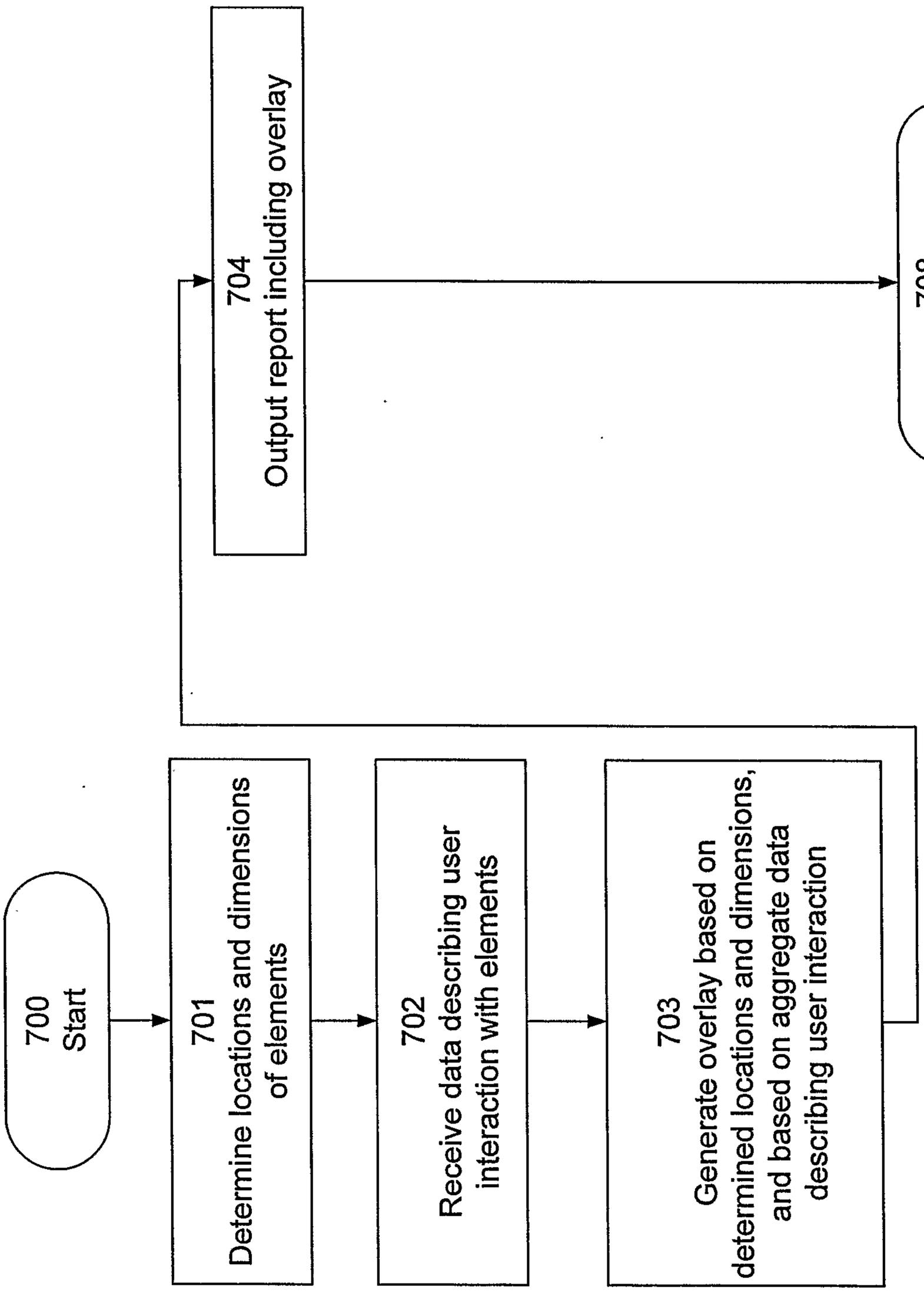
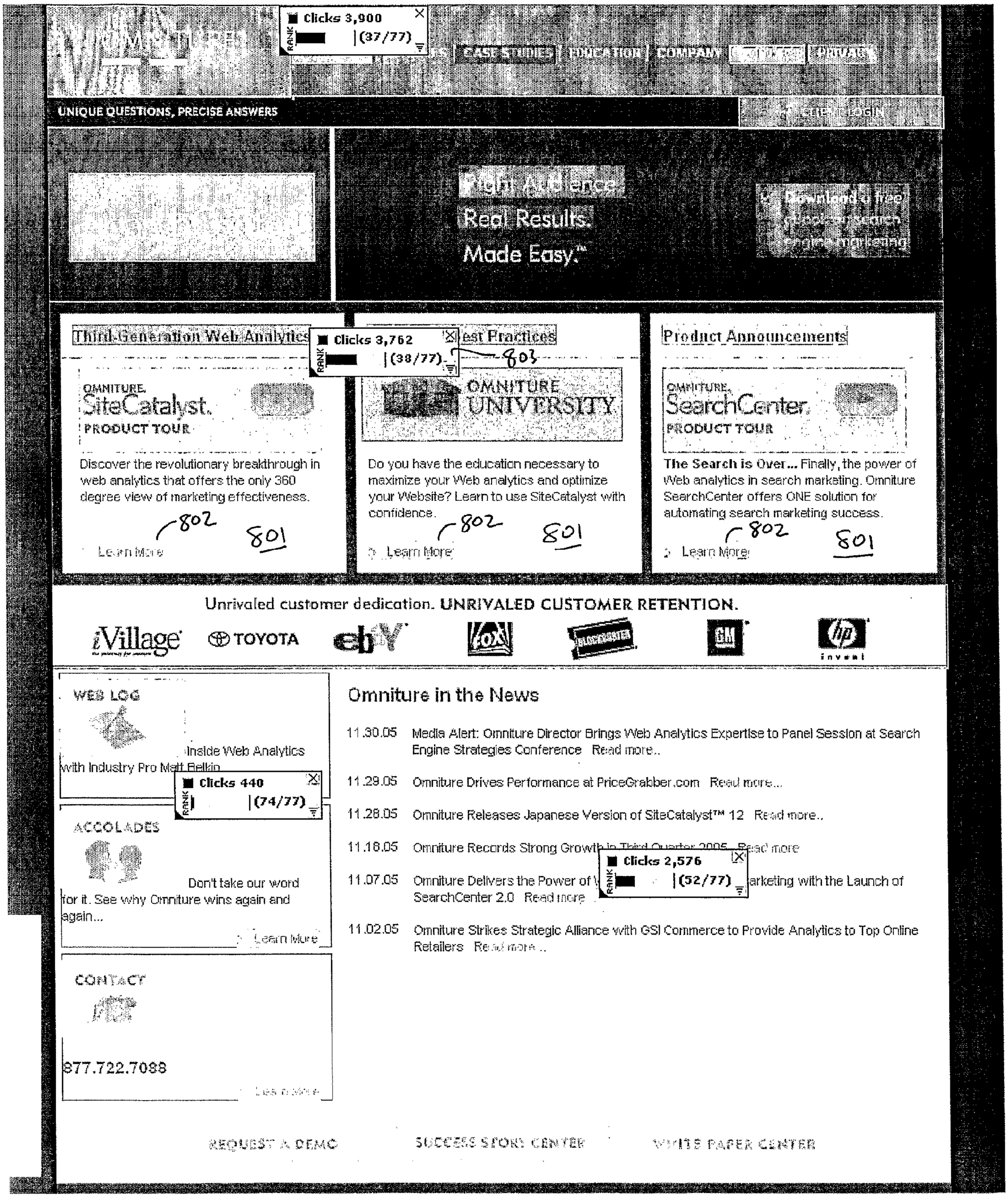
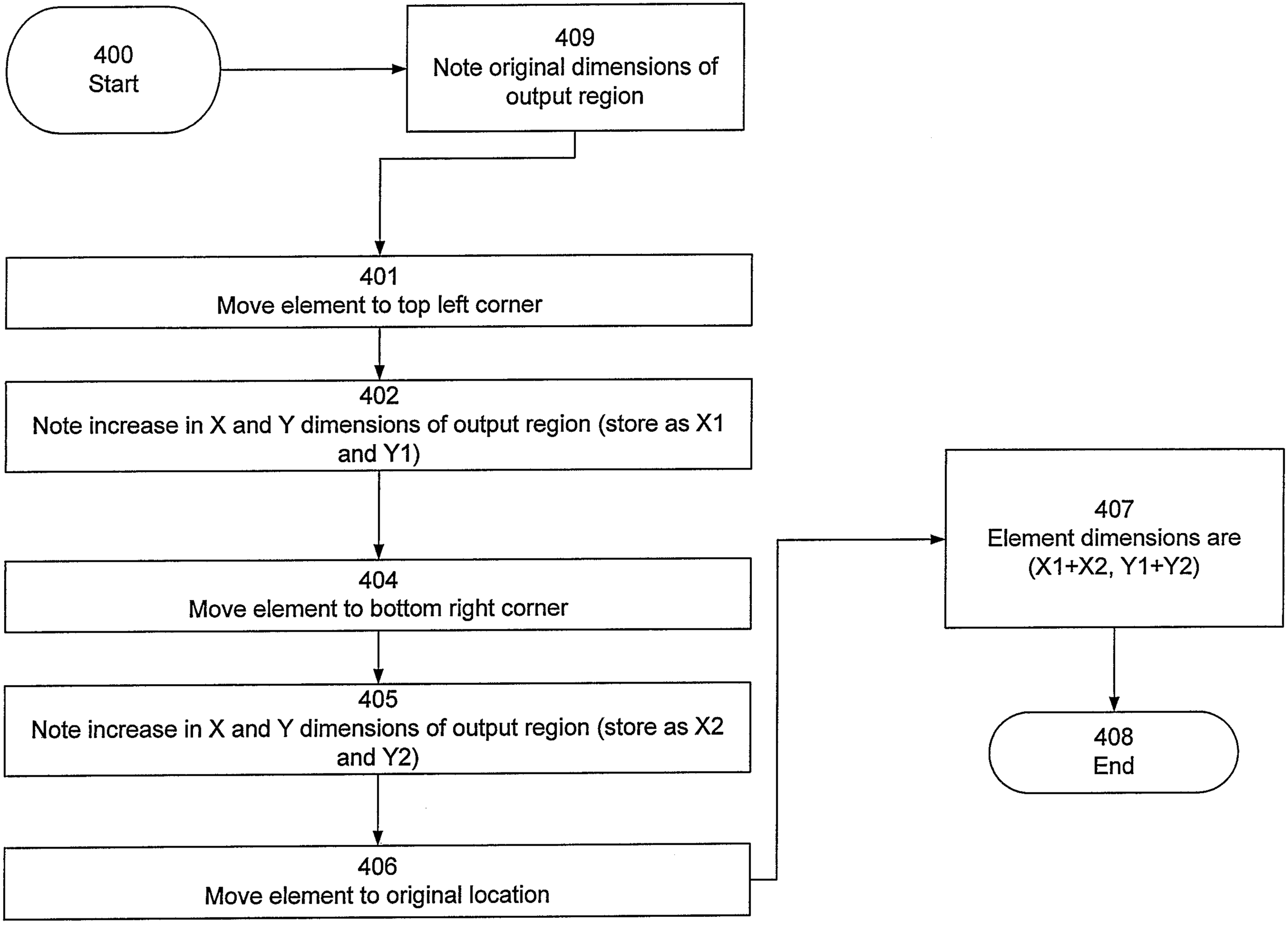

700
Start
701
Determine locations and dimensions of elements
702
Receive data describing user interaction with elements
703
Generate overlay based on determined locations and dimensions, and based on aggregate data describing user interaction
704
Output report including overlay
708
End

FIG. 7


Clicks 3,900
RANK (37/77)
CASE STUDIES
EDUCATION
COMPANY
PRIVACY
UNIQUE QUESTIONS, PRECISE ANSWERS
Right Audience.
Real Results.
Made Easy.™
Third-Generation Web Analytics
Clicks 3,762
RANK (38/77)
803
Best Practices
Product Announcements
OMNITURE SiteCatalyst. PRODUCT TOUR
Discover the revolutionary breakthrough in web analytics that offers the only 360 degree view of marketing effectiveness.
802
801
Learn More
OMNITURE UNIVERSITY
Do you have the education necessary to maximize your Web analytics and optimize your Website? Learn to use SiteCatalyst with confidence.
802
801
Learn More
OMNITURE SearchCenter. PRODUCT TOUR
The Search is Over... Finally, the power of Web analytics in search marketing. Omniture SearchCenter offers ONE solution for automating search marketing success.
802
801
Learn More
Unrivaled customer dedication. UNRIVALED CUSTOMER RETENTION.
eVillage
TOYOTA
ebay
FOX
BLOCKBUSTER
GM
hp invent
WEB LOG
Inside Web Analytics with Industry Pro Matt Belkin
Clicks 440
RANK (74/77)
ACCOLADES
Don't take our word for it. See why Omniture wins again and again...
Learn More
CONTACT
877.722.7088
Omniture in the News
11.30.05 Media Alert: Omniture Director Brings Web Analytics Expertise to Panel Session at Search Engine Strategies Conference Read more..
11.29.05 Omniture Drives Performance at PriceGrabber.com Read more...
11.28.05 Omniture Releases Japanese Version of SiteCatalyst™ 12 Read more..
11.18.05 Omniture Records Strong Growth in Third Quarter 2005 Read more
Clicks 2,576
RANK (52/77)
11.07.05 Omniture Delivers the Power of Web Analytics to Search Marketing with the Launch of SearchCenter 2.0 Read more
11.02.05 Omniture Strikes Strategic Alliance with GSI Commerce to Provide Analytics to Top Online Retailers Read more..
REQUEST A DEMO
SUCCESS STORY CENTER
WHITE PAPER CENTER

800

FIG. 8


400
Start
409
Note original dimensions of output region
401
Move element to top left corner
402
Note increase in X and Y dimensions of output region (store as X1 and Y1)
404
Move element to bottom right corner
405
Note increase in X and Y dimensions of output region (store as X2 and Y2)
406
Move element to original location
407
Element dimensions are (X1+X2, Y1+Y2)
408
End