



- (51) **International Patent Classification:**
G06T 13/00 (2011.01) *G06F 9/44* (2006.01)
- (21) **International Application Number:**
PCT/US2011/063458
- (22) **International Filing Date:**
6 December 2011 (06.12.2011)
- (25) **Filing Language:** English
- (26) **Publication Language:** English
- (30) **Priority Data:**
12/966,787 13 December 2010 (13.12.2010) US
- (71) **Applicant (for all designated States except US):** **MICROSOFT CORPORATION** [US/US]; One Microsoft Way, Redmond, Washington 98052-6399 (US).
- (72) **Inventors:** **LAU, Bonny**; c/o Microsoft Corporation, LCA - International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). **ZOU, Song**; c/o Microsoft Corporation, LCA - International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). **ZHANG, Wei**; c/o Microsoft Corporation, LCA - International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). **BECK, Brian**; c/o Microsoft Corporation, LCA - International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). **GLEASMAN, Jonathan**; c/o Microsoft Corporation, LCA - International

Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). **CHEN, Pai-Hung**; c/o Microsoft Corporation, LCA - International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US).

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

- (84) **Designated States** (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

[Continued on next page]

- (54) **Title:** COORDINATION OF ANIMATIONS ACROSS MULTIPLE APPLICATIONS OR PROCESSES

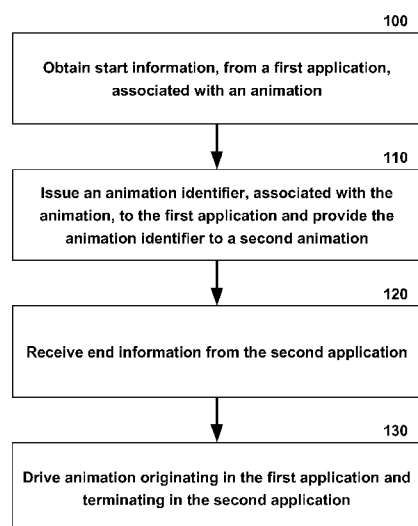


FIG. 1

- (57) **Abstract:** Animation coordination system and methods are provided that manage animation context transitions between and/or among multiple applications. A global coordinator can obtain initial information, such as initial graphical representations and object types, initial positions, etc., from initiator applications and final information, such as final graphical representations and object types, final positions, etc. from destination applications. The global coordination creates an animation context transition between initiator applications and destination applications based upon the initial information and the final information.

**Declarations under Rule 4.17:**

- *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*
- *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

- *with international search report (Art. 21(3))*
- *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

COORDINATION OF ANIMATIONS ACROSS MULTIPLE APPLICATIONS OR PROCESSES

TECHNICAL FIELD

5 **[0001]** The subject disclosure relates to coordination of animations across multiple applications or processes of a computing system, e.g., from one context of the computing system to another.

BACKGROUND

10 **[0002]** As available computing resources continue to increase in line with advances in computing technology, additional resources become expendable on providing rich user experiences. One mechanism to provide rich experiences is animation. Animations can be employed to create fluid and seamless transitions between different contexts. For instance, context transitions, e.g., a transition from one mode of an application to another mode of the application, can be animated to provide a user with a
15 seamless, visual switch between the modes.

20 **[0003]** However, in some situations, context transitions can involve or extend across multiple processes or applications. In an example of such a transition, opening a picture, in a picture viewing application, from a thumbnail representation in a file exploration application, involves a context switch from the file exploration application to the picture viewing application. Further to this example, the picture viewing application can display the picture in a full screen representation.

25 **[0004]** Typically, animating such a context switch, e.g., between the file exploration application and the picture viewing application, can be attempted in a unilateral or bilateral fashion. In a unilateral manner, either the file exploration application or the picture viewing application can drive a visual transition animation. However, the file exploration application, if it draws the animation, does not know where the destination point, e.g., of the display, is in the picture viewing application. Similarly, when the picture viewing application draws the animation, the picture viewing application does not possess knowledge on an origination point, in the file exploration application, of
30 the animation.

35 **[0005]** For bilaterally driven animations, applications involved in a transition will jointly draw an animation. According to the previous example, the file exploration application and the picture viewing application can both draw the animation. For instance, the file exploration can initiate the animation, e.g., start rendering, e.g., drawing, the

animation, and the picture viewing application can finish the animation. Accordingly, a handoff, of drawing responsibility, occurs between the file exploration application and the picture viewing application. The handoff causes a seam or incongruency in the animation, which is visible to the user.

5 **[0006]** The above-described deficiencies of today's visual transition systems are merely intended to provide an overview of some of the problems of conventional systems, and are not intended to be exhaustive. Other problems with conventional systems and corresponding benefits of the various non-limiting embodiments described herein may become further apparent upon review of the following description.

10

SUMMARY

[0007] A simplified summary is provided herein to help enable a basic or general understanding of various aspects of exemplary, non-limiting embodiments that follow in the more detailed description and the accompanying drawings. This summary is not intended, however, as an extensive or exhaustive overview. Instead, the sole purpose of
15 this summary is to present some concepts related to some exemplary non-limiting embodiments in a simplified form as a prelude to the more detailed description of the various embodiments that follow.

[0008] In one or more embodiments, a generic framework can be employed to create seamless animations of context transitions between arbitrary applications. The
20 framework includes an animation coordinator that interacts with applications to obtain initial information from initiator applications, e.g., applications at starting points of animations, and final information from destination applications, e.g., applications at ending points of animations. The animation coordinator generates animations based upon the initial information and the final animation and drives the animations from outside of
25 initiator applications and/or destination applications to enable seamless and smooth animated transitions between different contexts.

[0009] In some embodiments, the animation coordinator synchronizes initiator applications and destination applications such that the destination applications can immediately capture focus and/or control upon completion of the animation. For instance,
30 in an application launch transition, the animation coordinator can pause for an indication from a destination application that it is in a ready state. Upon receiving the indication, the animation coordinator can drive the animation.

[0010] These and other embodiments are described in more detail below.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] Various non-limiting embodiments are further described with reference to the accompanying drawings in which:

[0012] Figure 1 is a flow diagram illustrating an exemplary, non-limiting embodiment for coordinating an animation spanning across multiple applications;

[0013] Figure 2 is a flow diagram illustrating another exemplary and non-limiting embodiment for coordinating an animation across multiple applications;

[0014] Figure 3 is a flow diagram illustrating an exemplary, non-limiting embodiment for generating an animation;

[0015] Figure 4 is a flow diagram illustrating an exemplary, non-limiting embodiment for drawing an animation;

[0016] Figure 5 is an exemplary, non-limiting illustration of a transition animation that bridges multiple applications;

[0017] Figure 6 is a block diagram illustrating an exemplary, non-limiting animation coordination system;

[0018] Figure 7 is an exemplary, non-limiting illustration of a transition animation that bridges multiple applications and includes disparate initial and final graphical representations;

[0019] Figure 8 is a flow diagram illustrating an exemplary, non-limiting embodiment for coordinating one or more animations initiated by one application and terminating in a plurality of applications;

[0020] Figure 9 is an exemplary, non-limiting illustration of multiple transition animations originating in one application and terminating in a plurality of applications;

[0021] Figure 10 is a block diagram of an exemplary, non-limiting animation coordination system in accordance with one or more embodiments;

[0022] Figure 11 is a block diagram of an exemplary, non-limiting animation coordinator in accordance with one or more embodiment;

[0023] Figure 12 is block diagram of illustrating an exemplary, non-limiting animation coordination system in accordance with one or more embodiments;

[0024] Figure 13 is a block diagram representing exemplary non-limiting networked environments in which various embodiments described herein can be implemented; and

[0025] Figure 14 is a block diagram representing an exemplary non-limiting computing system, device, or operating environment in which one or more aspects of various embodiments described herein can be implemented.

DETAILED DESCRIPTION

5 OVERVIEW

[0026] As discussed in the background, visually animated context transitions between different applications cannot be generally employable by virtually any application on a computing system, e.g., a computing device, while remaining seamless to a user. For instance, two unrelated applications, e.g., applications developed and/or controlled by different developers, cannot exchange information to enable a seamless animated transition therebetween. While two applications, developed and/or controlled by a single developer, can be hard-coded with mechanisms to animate a context transition, such mechanisms are only available to and employable by those two applications for animation of transitions merely between the two applications.

10 [0027] Further, two arbitrary applications or processes, e.g., unrelated applications, could leverage other conventional mechanisms to implement a transitional animation. For instance, such applications can employ inter-process communication techniques, shared memory spaces, inter-process synchronization techniques, etc., to execute a handoff of a transitional animation commenced by one application and concluded by another application. Conventional mechanisms, however, cannot create a seamless and smooth animation as the handover point, e.g., the point where responsibility for drawing the animation changes, results in an animation artifact visible to a user. Accordingly, conventional techniques to create visual animations of context transitions cannot generate seamless animations between multiple arbitrary applications and processes in a generalized and generic manner.

20 [0028] In various, non-limiting embodiments, a generic framework can be employed to create seamless animations of context transitions between arbitrary applications. The generic framework provides a uniform application program interface (API) employable by applications to serve up information to create transitional animations. The generic framework retains information from initiator applications, e.g., applications from which animations originate, and destination applications, e.g., applications where animations terminate, and correlates such information to enable an animation to be drawn from a particular initiator application to a particular destination application.

[0029] According to further embodiments, the framework includes a global coordinator that interacts with the applications to obtain initial information and terminal information. In addition, the global coordinator controls animation timing and drawing, based upon the initial and terminal information, to create a seamless transition. Further, the global coordinator can communicate with coordinators, compositors, or other components of other virtual or physical machines to animate context switches between applications and/or processes on disparate virtual or physical machines.

[0030] In one embodiment, a method is described herein that includes receiving initial information from an initiator application, wherein the initial information relates to a transitional animation that visualizes a shift in user context from the initiator application to a destination application, receiving final information, associated with the transitional animation, from the destination application, generating the transitional animation based at least in part on the initial information and the final information, and drawing the transitional animation on a display device of the computing system. The method can also include generating an animation identifier that corresponds to the transitional animation, issuing the animation identifier to at least one of the initiator application or the destination application, and retaining the initial information and the final information in association with the animation identifier. Further, in an example, receiving the final information can include receiving the final information from the destination application along with the animation identifier to enable association of the final information with the initial information. In another example, generating the transitional animation can include creating a plurality of intermediate graphical representations that bridge a difference between an initial graphical representation included in the initial information and a final graphical representation specified in the final information, wherein a number of intermediate graphical representations created is in accordance with a predetermined frame rate of the transitional animation, and can further include determining a set of position information respectively associated with the plurality of intermediate graphical representations, wherein position information is determined based upon at least one of initial position information in the initial information, final position information included in the final information, or other position information included in the set of position information.

[0031] According to further examples, drawing the transitional animation can include ordering the initial graphical representation, the plurality of intermediate graphical representations, and the final graphical representation to create a frame order, and

sequentially and individually drawing the initial graphical representation, the plurality of intermediate graphical representations, and the final graphical representation in accordance with the frame order, wherein a time between drawings is determined based upon the predetermined frame rate. Moreover, drawing can involve enforcing a z-position of the transitional animation such that the transitional animation remains a top-most object on the display device. In yet another example, the method can include waiting for a readiness indicator from the destination application before commencing drawing of the transitional animation, wherein receiving the end information from the destination application is the readiness indicator.

[0032] In another embodiment, a system, as described herein, includes an animation coordination component which includes an input component configured to obtain initial information from one or more initiator applications and to obtain final information from one or more destination applications and an animation component configured to create and to draw animations on a display device, e.g., display screen, wherein the animations provide seamless transitions from environments associated with the one or more initiator applications to environments associated with the one or more destination applications. The animation coordination component can further include an identifier generation component configured to issue animation identifiers to individual animations coordinated by the animation coordination component, wherein the animation identifiers comprise global identifiers (GIDs), such as, but not limited to, globally unique identifiers (GUIDs). An output component is provided that is configured to transmit animation identifiers to the one or more initiator applications as a receipt of the initial information. In further examples, the animation coordination component can include a frame generation component configured to create animation frames for the animations to be drawn by the animation component.

[0033] The initial information, in an example, includes at least one of initial graphical representations of objects in the one or more initiator applications, types of the objects in the one or more initiator applications, respective positions of the initial graphical representations, or respective sizes of the initial graphical representations.

Similarly, the final information includes at least one of final graphical representations of objects in the one or more destination applications, types of the objects in the one or more destination applications, respective positions of the final graphical representations, or respective sizes of the final graphical representations.

[0034] In an additional embodiment, an animation coordination subsystem can be integrated into an operating system and include a first application program interface (API) configured to enable triggering of an animation that seamlessly visualizes a transition between disparate contexts, wherein the first API is adapted to receive initial information, from an initiator application, that represents a starting point for the animation and to return an animation identifier corresponding to the animation and a second API configured to receive final information that represents an ending point for the animation, wherein the final information is received in connection with the animation identifier and from a destination application. The animation coordination module generates the animation based upon the initial information and the final information and outputs the animation to a display screen.

[0035] Herein, an overview of some of the embodiments for providing seamless transitional animations across multiples applications has been presented above. As a roadmap for what follows next, various exemplary, non-limiting embodiments and features for global animation coordination are described in more detail. Then, some non-limiting implementations and examples are given for additional illustration, followed by representative network and computing environments in which such embodiments and/or features can be implemented.

TRANSITIONAL ANIMATIONS BETWEEN MULTIPLE APPLICATIONS/ PROCESSES

[0036] As mentioned above, in various embodiments, a framework is provided which is configured to generate animations that extend between multiple applications. In one example, the animations include a visual morph of a graphical representation of an object in a first application to another graphical representation of an object in a second application. The initial graphical representation and the terminal graphical representation, in one aspect, can be different representations of a common base object. For instance, the initial graphical representation and terminal graphical representation can vary in one or more characteristics, e.g., resolution, size, fidelity, granularity, location, etc., and the animation can comprise a seamless transformation from the initial graphical representation to the terminal graphical representation. In another aspect, the initial graphical representation and the terminal graphical representation can represent disparate objects such as but not limited to, text objects, picture objects, image objects, video objects, map objects, list objects, web objects, or the like. In the case of disparate objects, the animation can comprise a morphological transformation from one representation of one object to another representation of another object. For instance, the animation can visually

depict a transformation of a portion of text into an image. In further embodiments, animations can be generated that transform a plurality of objects into a single object and/or that transform a single object into a plurality of objects.

[0037] Moreover, the multiple applications can reside on a single virtual or physical machine or reside on different virtual and/or physical machines. Further, the multiple applications can be multiple instances or processes of a single application. In yet another embodiment, a single application can leverage the framework provided herein for intra-application transitions. For instance, a browser application can utilize animated transitions when browsing between different domains, e.g., browsing from a first web page to a second web page. Accordingly, the framework, in addition to creating inter-application animations, can be leveraged for intra-application transitions, in a generalized manner.

[0038] With respect to one or more non-limiting ways to coordinate animations of context transitions as described above, Fig. 1 shows a flow diagram illustrating an exemplary, non-limiting embodiment for coordinating an animation spanning across multiple applications. At 100, start information associated with an animation is obtained from a first application. In a specific, non-limiting example, the start information is obtained by a global coordinator, as described herein, which can reside on a physical or virtual machine along with the first application. However, it will be appreciated that the global coordinator can reside on a different physical or virtual machine from the first application.

[0039] The start information can specify parameters to commence an animation. For instance, the start information can include an initial graphical representation of an object, in the first application, which is a starting frame of the animation. The start information can also include a position, e.g., x-coordinates, y-coordinates, etc., on a display at which the initial graphical representation is drawn. The position on the display can relate to a position on a real, physical display or a position on a virtual display. Further, a size of the initial graphical representation can be included in the start information. In an example, the position and size information can be conveyed as a bounding rectangle of the initial graphical representation. The bounding rectangle can be specified by a height, a width, and coordinates of a pre-determined corner, e.g., upper-left, upper-right, lower-left, or lower-right.

[0040] At 110, an animation identifier, associated with the animation, can be issued. In one aspect, the animation identifier can be a global identifier; however, other

types of identifiers can be employed as the animation identifier. According to one embodiment, the global coordinator can generate the animation identifier and issue the animation identifier to the first application. In another embodiment, the first application can generate the animation identifier. In addition, the global coordinator and/or the first application, depending on which generates the identifier, can provide the animation identifier to a second application where the animation concludes.

[0041] At 120, end information is received from the second application. End information, similar to the start information, can include a final graphical representation, of the second application, which is a final frame of the animation. In addition, the end information can include a position of the final graphical representation, a size of the final graphical representation, and/or a bounding rectangle of the final graphical representation. At 130, the animation, originating in the first application and terminating in the second application, is driven based upon the start information and the end information. For instance, the global coordinator can draw the animation beginning with the initial graphical representation at the position indicated in the start information and ending with the final graphical representation at the position indicated in the end information. The global coordinator can influence a z-order of graphical objects, e.g., windows, etc., on the display to ensure that the animation remains the top-most object relative to at least the first application and the second application.

[0042] In a further embodiment, the animation can be driven in accordance with timing or synchronization enforced by the global coordinator. For instance, the global coordinator starts rendering the animation after receiving notification that the second application is ready. The notification can be explicit, e.g., the second application sends a specific message to the global coordinator, or the notification can be implicit, e.g., the global coordinator presumes readiness upon receiving the end information. Prior to sending notification of readiness, the second application can render an associated window and user interface, but keep the user interface hidden. Once the animation completes, the rendered window and user interface can be made visible to create a seamless context transition. By waiting for a readiness signal from the second application, the seamless animation can occur since the user interface of the second application can be displayed quickly.

[0043] Fig. 2 shows a flow diagram illustrating another exemplary and non-limiting embodiment for coordinating an animation spanning across multiple applications. At 200, initial information from an initiator application is received. The initial

information relates to and triggers a transitional animation that visualizes a shift in user context from the initiator application to a destination application. At 210, an animation identifier, corresponding to the transitional animation, is generated. At 220, the animation identifier is issued to the initiator application and/or the destination application. The
5 animation identifier provides a receipt of the initial information as well as indicates that the transitional animation is being constructed. Moreover, the animation identifier provides a reference which can be employed to associate various data and/or messages to the transitional animation.

[0044] At 230, final information, associated with the transitional animation, is
10 received from the destination application. In one example, the final information can be received in connection with the animation identifier. At 240, the initial information and the final information can optionally be stored in association with the animation identifier. In a further embodiment, prior to commencing display of the transitional animation, the process can wait until a readiness indicator is received from the destination application. In
15 a specific, non-limiting example, receiving the final information can operate as an implicit readiness indication.

[0045] At 250, the transitional animation is generated. Turning briefly to Fig. 3, illustrated is a flow diagram of an exemplary, non-limiting embodiment for generating an animation. At 300, a plurality of intermediate graphical representations can be created.
20 The intermediate graphical representations can bridge a gap between an initial graphical representation and a final graphical representation. In an embodiment, the number of intermediate graphical representations created can be based upon a predetermined frame rate of the transitional animation. At 310, a set of positions are determined, wherein the set of positions are respectively associated with the plurality of intermediate graphical
25 representations. According to an example, the positions can be determined based upon an initial position of the initial graphical representation, a final position of the final graphical representation, and/or other positions in the set of positions. At 320, animation frames are generated by matching graphical representations to positions.

[0046] Turning back to Fig. 2, the transitional animation is drawn at 260, which is
30 further illustrated in an exemplary, non-limiting embodiment depicted in Fig. 4. As shown in Fig. 4, at 400, a set of graphical representations are ordered to create a frame order. At 410, graphical representations from the set of graphical representations are sequentially and individually drawn in accordance with the frame order. Additionally, at 420, during

drawing, a z-position of the animation is enforced to maintain the animation at a top-most position.

[0047] Fig. 5 is an exemplary, non-limiting illustration of a transition animation that bridges multiple applications. As shown in Fig. 5, an animation can visually perform a context transition from a first application, e.g., email application 500, to a second application, e.g., picture viewer application 510. Email application 500 can be any suitable email application capable of receiving, viewing, drafting, sending, etc., emails. As illustrated in Fig. 5, email application 500 can view emails which contain embedded images, such as image 502. In one embodiment, email application 500 displays image 502 as a thumbnail representation embedded within a displayed email message.

[0048] A user can elect to view a larger representation, if available, in picture viewer application 510. To do so, the user can double-click image 502, right-click image 502 and select a command to open image 502 in picture viewer application 510, or employ some other means available in the user interface of email application 500. In a non-limiting example, email application 500 can utilize facilities of a host operating system to look up file associations maintained by the operating system to identify which application is associated with a file type of image 502. Once email application 500 identifies picture viewer application 510 as the application associated with the file type of image 502, email application 500 can employ additional mechanisms of the operating system to launch picture viewer application 510 with a file corresponding to image 502. Picture viewer application 510 can execute, open the file corresponding to image 502, and display a larger representation in the form of image 512. In another example, email application 500 can signal to the operating system a file type and/or file name desired to be opened/viewed in a separate application. The operating system can identify the appropriate program, e.g., picture viewer application 510, execute the program, and provide the program with the file.

[0049] An application launch scenario, as described in the above example, involves a sudden context transition from the environment of email application 500 to the environment of picture viewer application 510. Typically, such a context transition is abrupt. For instance, the user double-clicks image 502 in email application 500 and instructs the operating system to launch picture viewer application 510 with the file associated with image 502. Once picture viewer application 510 commences execution, various background tasks are performed. Picture viewer application 510 can access the file and access stored information of the file. Further, picture viewer application 510 can

display a splash screen during background processing. Ultimately, picture viewer application 510 renders a window and/or a user interface which is displayed on a screen or other output display. Conventionally, the transition from double-clicking image 502 to displaying the window or user interface of picture viewer application 510 can be sharp.

5 For instance, from a visual perspective, e.g., what the user sees on the output display, the user views email application 500, followed by picture viewer application 510 becoming visible in an instant after some time period passes. Optionally, display of picture viewer application 510 can be preceded by an abrupt display of a splash screen. In another example, depending on system resources and/or a manner in which the application is
10 coded, the display of picture viewer application 510 can be discontinuous with various portions of the user interface becoming visible in a stepped fashion.

[0050] As described above, an animation can be generated to provide a seamless and smooth transition from email application 500 to picture viewer application 510. The animation, as shown in Fig. 5, can be a visual enlargement or conversion of image 502,
15 e.g., an initial graphical representation, to image 512, e.g., a final graphical representation. The animation can include a plurality of frames ordered from a start frame (image 502) to an end frame (image 512). Intermediate frames, such as frame 504 and frame 514 can be generated. Intermediate frames can be generated from a preceding frame and/or the start frame. In another example, intermediate frames can be generated in a reverse order, e.g.,
20 based upon a subsequent frame and/or the end frame. It will be appreciated that both generation techniques can be employed to provide high quality intermediate frames closely related to neighboring frames. For instance, an intermediate frame close to the end frame can have higher quality if generated from the end frame as opposed to the start frame. A number of intermediate frames can depend on various factors such as, but not
25 limited to, duration of the animation and a desired frame rate. According to a non-limiting example, consider a two second animation with a frame rate of 24 frames per second. The animation would include 48 total frames. The frames can be ordered and displayed such that image 502, occupying a start position, morphs into a larger representation, e.g., image 512 while translating to a final position occupied by image 512.

30 **[0051]** Fig. 6 is a block diagram illustrating an exemplary, non-limiting animation coordination system. The animation coordination system includes an application 600, e.g., a first application, and an application 610, e.g., a second application, between which an animated context transition occurs. The system can also include a coordinator 620 configured to generate an animation 624, synchronize application 600 and application 610,

and draw animation 624. In a specific, non-limiting embodiment, coordinator 620 can be a global component of an operating system (not shown) hosting application 600 and application 610. However, it will be appreciated that coordinator 620 can be a separate process from the operating system.

5 **[0052]** Application 600 can initiate animation 624 by sending start information 602 to coordinator 620. Start information 602, in an embodiment, can specify one or more initial graphical representations, initial position information, and/or initial size information. As described above, coordinator 620 can generate animations that transition many objects into a single object, a single object into many objects, and/or a single object
10 into another single object. Accordingly, start information 602 can include a plurality of initial graphical representations with respective position and size information in the case of a many-to-one transition. In the case of a one-to-many or a one-to-one transition, start information 602 can include a single initial graphical representation with respective position and size information.

15 **[0053]** Coordinator 620, upon receipt of start information 602, generates a global identifier (GID) 622 as an animation identifier. GID 622 is associated with animation 624 and employed to index start information 602 as related to animation 624. Coordinator 620 provides GID 622 to application 600 in response to start information 602. In addition, coordinator 620 can also provide GID 622 to application 610. Alternatively, application
20 600 can include GID 622 in application launch information passed to application 610 via operating system launch mechanisms employed to start application 610 from application 600. In either regard, GID 622, when received by application 610, operates to notify application 610 that animation 624 will occur to visually transition from a context of application 600 to a context of application 610.

25 **[0054]** Upon obtaining GID 622, application 610 can transmit end information 612 to coordinator 620. End information 612, similar to start information 602, can include one or more final graphical representations, final position information, and/or final size information. Application 610 can transmit end information 612 in connection with GID 622 to enable coordinator 620 to associate end information 612 with start information 602
30 and animation 624 already associated with GID 622.

[0055] Application 610 can transmit end information 612 as an implicit indication of readiness. For instance, by sending end information 612, application 610 notifies coordinator 620 that application 610 is prepared to catch animation 624 when drawn. Accordingly, application 610 can perform any start-up processing to prepare to display a

user interface or window, prior to transmitting end information 612, to prevent animation 624 completing before application 610 can draw its window or user interface.

[0056] Upon receiving end information 612, coordinator 620 can generate animation 624 based, at least in part, on start information 602 and end information 612.

5 For instance, coordinator 620 can utilize initial graphical representations in start information 602 as start frames and final graphical representations in end information 612 as end frames. Coordinator 620 generates a plurality of intermediate frames from the initial graphical representations and/or the final graphical representations as described in connection with Fig. 5. Coordinator 620 can order and draw frames of animation 624 to
10 provide a seamless and smooth transition from the context of application 600 to the context of application 610.

[0057] As described above, the graphical representations, which are end points of animations, can depict a variety of objects, such as, but not limited to, lists of information, pictures, photos, clip art, logos, text, videos, web pages, graphics, maps, combinations of
15 the foregoing, etc. For illustrative purposes, the following non-exhaustive and non-limiting examples reveal some types of animations that can be employed for context transitions between applications, within applications, between virtual machines, between physical machines, etc. In one example, a barcode can be an initial graphical representation for animation that terminates at some representation of a product specified
20 by the barcode. For instance, the barcode can transition to a web page about the product, an image of the product, a specification sheet on the product, or the like. Similarly, an advertisement or image associated with a company, e.g., on a web page, in an electronic magazine, etc., can transition into a product list of offerings from the company, a web page to purchase a product, or the like. In another example, a contact name, e.g., in an
25 address book application, on an electronic business card, etc., can be animated into another representation of the person, such as a photograph, a public social media profile, a map depicting a location, etc. In yet another example, a diagram or illustration of an object can be animated from a collapsed version to an exploded version. For instance, an engineering design application can employ embodiments described herein to animate an explosion of a
30 design of a device.

[0058] In a further example, Fig. 7 is an exemplary, non-limiting illustration of a transition animation that bridges multiple applications and includes initial and final graphical representations associated with disparate objects. As shown in Fig. 7, an animation can visually perform a context transition from a first application, e.g., email

application 700, to a second application, e.g., map viewer application 710. In particular, the animation can involve a contextual transition from an address 702 in an email message to a corresponding location on a map 712. Launching of map viewer application 710 from email application 700 can occur in a manner similar to that described above with reference to Fig. 5. In addition, coordination of the animation between email application 700 and map view application 710 can be implemented with the methods and systems previously described.

[0059] The initial graphical representation for the exemplary animation illustrated in Fig. 7 is address 702, which is made up of a portion of text bounded by an invisible rectangle. The final graphical representation is map 712, which comprises a graphical image of roads, intersections, buildings, satellite imagery, etc. A coordination component, such as coordinator 620, configured to generate and control animations bridging multiple applications, can generate one or more intermediate frames 704. The intermediate frames 704 can be based upon various composites of the initial graphical representation and the final graphical representations to provide an animated metamorphosis from the text included in address 702 to the graphics included in map 712.

[0060] In a further embodiment, Fig. 8 illustrates a flow diagram of an exemplary, non-limiting method for coordinating one or more animations initiated by one application and terminating in a plurality of applications. At 800, start information associated with one or more animations is obtained from a first application. At 810, an animation identifier is generated and issued to the first application, wherein the animation identifier is associated with the one or more animations. Further, the animation identifier is provided to a set of destination applications. In one embodiment, a single animation identifier can be issued collectively for the one or more animations individually commencing from an initial graphical representation specified in the start information and respectively terminating at end points in one or more applications in the set of destination applications. In another example, distinct animation identifiers can be generated and issued for each pairing of the first application to a destination application in the set.

[0061] At 820, respective end information is received from one or more destination applications in the set of destination applications. The end information can collectively include final graphical representations for each of the one or more destination applications as well as respective position and size information for each final graphical representation. At 830, one or more animations are drawn, initiating from the first

application and terminating in the one more destination applications from the set of destination applications.

[0062] Fig. 9 is an exemplary, non-limiting illustration of multiple transition animations originating in one application and terminating in a plurality of applications. As shown in Fig. 9, an animation can visually perform a context transition from a first application, e.g., email application 900, to a plurality of destination applications, e.g., address book application 910 and browser application 920. In particular, Fig. 9 illustrates a plurality of animations respectively depicting a context transition from email application 900 to address book application 910 and a context transition from email application 900 to browser application 920. The context switch between email application 900 and address book application 910 can involve viewing an entry in an address book associated with contact 902, e.g., name of contact, included in an email message displayed by email application 900. Additionally, the context switch between email application 900 and browser application 920 can involve a transition from selecting contact 902 in the email message to viewing/displaying a public social media profile in browser application 920.

[0063] The initial graphical representation for the example animations illustrated in Fig. 9 is contact 902, which is made up of a portion of text bounded by an invisible rectangle. The final graphical representations of the animations include an address book entry 912 in the address book application and a profile 922 in browser application 920. As depicted in Fig. 9, address book entry 912 can comprise a combination of images and text. Similarly, profile 922 can be a web page that likewise includes a variety of media such as text, images, videos, etc. To generate the animations, intermediate frames can be respectively generated for each constituent animation. For instance, intermediate frames 914 can be generated for the constituent animation between contact 902 and address book entry 912 and intermediate frames 924 are created for the constituent animation between contact 902 and profile 922.

[0064] In an embodiment, the constituent animations can be drawn jointly or separately. For instance, the animation between contact 902 and address book entry 912 can be drawn once address book application 910 is ready even if browser application 920 is not. In another example, neither animation commences until both address book application 910 and browser application 920 report readiness.

[0065] Fig. 10 is a block diagram of an exemplary, non-limiting animation coordination system in accordance with one or more embodiments. The animation coordination system includes an initiator application 1000 and a set of destination

applications 1010 between which animated context transitions occur. The system can also include a coordinator 1020 configured to generate animations 1024 depicting the context transitions between the initiator application 1000 and the set of destination applications 1010.

5 **[0066]** Initiator application 1000 can utilize animations 1024 to visualize context transitions with the set of destination applications 1010. To initiate an animation, initiator application 1000 sends start information 1002 to coordinator 1020. Start information 1002 can include an initial graphical representation, initial position information, and/or initial size information. Coordinator 1020, upon receipt of start information 1002, generates one
10 or more global identifiers (GIDs) 1022. GIDs 1022 are associated with animations 1024 and are employed to index start information 1002 as related to animations 1024. Coordinator 1020 provides GIDs 1022 to initiator application 1000 in response to start information 1002. In addition, coordinator 1020 can also provide GIDs 1022 to applications in the set of destination applications 1010.

15 **[0067]** Upon obtaining GIDs 1022, the set of destination applications 1010 can transmit a set of end information 1016 to coordinator 1020. The set of end information 1016 includes end information respectively associated with each destination application in the set of destination applications 1010. The set of destination applications 1010 can include one or more applications, such as destination application 1012 and destination
20 application 1014, which include graphical representations that are end points of transitional animations. While Fig. 10 depicts two destination applications in the set of destination applications 1010, it is to be appreciated that the set of destination applications 1010 can include N applications, where N is an integer greater than or equal to one. The set of end information 1016 can include a set of final graphical representation, a set of
25 final position information, and/or a set of final position information. One group consisting of a final graphical representation in the set, along with associated final position information and final size information, can correspond to one destination application in the set of destination applications 1010.

30 **[0068]** Upon receiving the set of end information 1016, coordinator 1020 can generate animations 1024 based on start information 1002 and the set of end information 1016. Coordinator 1020 can separately draw sub-animations, e.g., an animation between initiator application 1000 and one of the destination applications, or can jointly draw all animations between initiator application 1000 and the set of destination applications 1010.

[0069] With reference to Fig. 11, a block diagram is provided that illustrates an exemplary, non-limiting animation coordinator 1100 in accordance with one or more embodiments. As shown in Fig. 11, coordinator 1100 can include an input component 1102 configured to obtain start information, end information, readiness indicators, or the like, from one or more applications involved in an animated context transition. As described above, start information can include initial graphical representations, types of objects associated with the initial graphical representations, initial positions of the graphical representations, sizes of the graphical representations, identities, addresses, etc., of one or more destination applications at which animated transitions terminate, or any other suitable information to facilitate global rendering of an animation between applications. Similarly, end information specifies final graphical representations, types of objects associated with the final graphical representations, and positions and/or sizes of the graphical representations. In addition, end information can also include an animation identifier, associated with a particular animation, which correlates end information to corresponding start information. Input component 1102 can store start information and end information, in association with the animation identifier, in storage 1104.

[0070] An identifier generation component 1106 is provided, which is configured to generate the animation identifier for a new animation. For instance, upon receiving new start information from an application, coordinator 1100 commences a process to generate and draw a transitional animation. Coordinator 1100, in response to receiving start information, can employ identifier generation component 1106 to generate a corresponding animation identifier. In one embodiment, identifier generation component 1106 generates global identifiers (GIDs) to be employed as animation identifiers. However, it is to be appreciated that other types of identifiers can be utilized as animation identifiers in place of GIDs. Coordinator 1100, via an output component 1108, can provide the generated animation identifier to the application which sent the new start information. In addition, output component 1108 can send the animation identifier to any destination applications specified in the start information.

[0071] Once start information is received from an initiator application and end information is received from a destination application, coordinator 1100 can create a suitable animation based upon the received information. In one embodiment, coordinator 1100 can take the initial graphical representation and corresponding position/size information as a starting frame of the animation and the final graphical representation (and position/size details) as a final frame of the animation. Coordinator 1100 can employ a

frame generation component 1110 to create intermediate frames of the animation between the starting frame and the final frame. The frame generation component 1110 can perform various transforms on the final graphical representation and/or the initial graphical representation to create intermediate graphical representations illustrating a transitional point between the initial graphical representation and the final graphical representation. In addition, the frame generation component 1110 can determine suitable positional information for the intermediate graphical representations based upon the initial position information and the final position information. Coordinator 1100 can store generated intermediate frames in storage 1104 in association with the animation identifier.

Coordinator 1100 further includes an animation component 1112 configured to draw the starting frame, the intermediate frames, and the final frame on a display with an appropriate order and timing to create a seamless animation.

[0072] Fig. 12 is block diagram of illustrating an exemplary, non-limiting animation coordination system in accordance with one or more embodiments. As shown in Fig. 12, an animation coordination module 1202 can be integrated into an operating system 1200. Operating system 1200 can be provided within a computing system to manage hardware components (e.g., input devices, output devices, memory devices, storage devices, processing devices, etc.) of the computing system and provide common services (e.g., runtime libraries, etc.) to programs executed on the computing system, such as initiator and/or destination applications as described above. For instance, operating system 1200 can include a plurality of modules, such as animation coordination module 1202, extending common functionality via a plurality of application program interfaces (APIs). Operating system 1200 can receive a call to an API, execute a function or module associated therewith, and return a result to an entity initiating the call.

[0073] Animation coordination module 1202 can include a first API 1204 employable to trigger a transitional animation between arbitrary applications. First API 1204 is adapted to receive initial information, from an initiator application, and return an animation identifier. Animation coordination module 1202 further includes a second API 1206 configured to receive final information from a destination application. Animation coordination module 1202, upon receiving the final information via second API 1206, is configured to generate the transitional animation and output the animation to a display screen. For instance, animation coordination module 1202 can utilize a display driver (not shown) of operating system 1200 to access a video rendering device (e.g., graphics card in

the computing system hosting operating system 1200) to visually display the animation on the display screen coupled to the video rendering device.

EXEMPLARY NETWORKED AND DISTRIBUTED ENVIRONMENTS

[0074] One of ordinary skill in the art can appreciate that the various embodiments

5 of transitional animation system and methods described herein can be implemented in connection with any computer or other client or server device, which can be deployed as part of a computer network or in a distributed computing environment, and can be connected to any kind of data store. In this regard, the various embodiments described herein can be implemented in any computer system or environment having any number of
10 memory or storage units, and any number of applications and processes occurring across any number of storage units. This includes, but is not limited to, an environment with server computers and client computers deployed in a network environment or a distributed computing environment, having remote or local storage.

[0075] Distributed computing provides sharing of computer resources and services

15 by communicative exchange among computing devices and systems. These resources and services include the exchange of information, cache storage and disk storage for objects, such as files. These resources and services also include the sharing of processing power across multiple processing units for load balancing, expansion of resources, specialization of processing, and the like. Distributed computing takes advantage of network
20 connectivity, allowing clients to leverage their collective power to benefit the entire enterprise. In this regard, a variety of devices may have applications, objects or resources that may participate in the resource management mechanisms as described for various embodiments of the subject disclosure.

[0076] Fig. 13 provides a schematic diagram of an exemplary networked or

25 distributed computing environment. The distributed computing environment comprises computing objects 1310, 1312, etc. and computing objects or devices 1320, 1322, 1324, 1326, 1328, etc., which may include programs, methods, data stores, programmable logic, etc., as represented by applications 1330, 1332, 1334, 1336, 1338. It can be appreciated that computing objects 1310, 1312, etc. and computing objects or devices 1320, 1322,
30 1324, 1326, 1328, etc. may comprise different devices, such as personal digital assistants (PDAs), audio/video devices, mobile phones, MP3 players, personal computers, laptops, etc.

[0077] Each computing object 1310, 1312, etc. and computing objects or devices 1320, 1322, 1324, 1326, 1328, etc. can communicate with one or more other computing

objects 1310, 1312, etc. and computing objects or devices 1320, 1322, 1324, 1326, 1328, etc. by way of the communications network 1340, either directly or indirectly. Even though illustrated as a single element in Fig. 13, communications network 1340 may comprise other computing objects and computing devices that provide services to the system of Fig. 13, and/or may represent multiple interconnected networks, which are not shown. Each computing object 1310, 1312, etc. or computing object or device 1320, 1322, 1324, 1326, 1328, etc. can also contain an application, such as applications 1330, 1332, 1334, 1336, 1338, that might make use of an API, or other object, software, firmware and/or hardware, suitable for communication with or implementation of the application testing techniques provided in accordance with various embodiments of the subject disclosure.

[0078] There are a variety of systems, components, and network configurations that support distributed computing environments. For example, computing systems can be connected together by wired or wireless systems, by local networks or widely distributed networks. Currently, many networks are coupled to the Internet, which provides an infrastructure for widely distributed computing and encompasses many different networks, though any network infrastructure can be used for exemplary communications made incident to the systems as described in various embodiments.

[0079] Thus, a host of network topologies and network infrastructures, such as client/server, peer-to-peer, or hybrid architectures, can be utilized. The “client” is a member of a class or group that uses the services of another class or group to which it is not related. A client can be a process, i.e., roughly a set of instructions or tasks, that requests a service provided by another program or process. The client process utilizes the requested service without having to “know” any working details about the other program or the service itself.

[0080] In a client/server architecture, particularly a networked system, a client is usually a computer that accesses shared network resources provided by another computer, e.g., a server. In the illustration of Fig. 13, as a non-limiting example, computing objects or devices 1320, 1322, 1324, 1326, 1328, etc. can be thought of as clients and computing objects 1310, 1312, etc. can be thought of as servers where computing objects 1310, 1312, etc., acting as servers provide data services, such as receiving data from client computing objects or devices 1320, 1322, 1324, 1326, 1328, etc., storing of data, processing of data, transmitting data to client computing objects or devices 1320, 1322, 1324, 1326, 1328,

etc., although any computer can be considered a client, a server, or both, depending on the circumstances.

[0081] A server is typically a remote computer system accessible over a remote or local network, such as the Internet or wireless network infrastructures. The client process
5 may be active in a first computer system, and the server process may be active in a second computer system, communicating with one another over a communications medium, thus providing distributed functionality and allowing multiple clients to take advantage of the information-gathering capabilities of the server.

[0082] In a network environment in which the communications network 1340 or
10 bus is the Internet, for example, the computing objects 1310, 1312, etc. can be Web servers with which other computing objects or devices 1320, 1322, 1324, 1326, 1328, etc. communicate via any of a number of known protocols, such as the hypertext transfer protocol (HTTP). Computing objects 1310, 1312, etc. acting as servers may also serve as clients, e.g., computing objects or devices 1320, 1322, 1324, 1326, 1328, etc., as may be
15 characteristic of a distributed computing environment.

EXEMPLARY COMPUTING DEVICE

[0083] As mentioned, advantageously, the techniques described herein can be applied to any device where it is desirable to create animations that visually depict context transitions between multiple arbitrary applications or processes in a computing system. It
20 can be understood, therefore, that handheld, portable and other computing devices and computing objects of all kinds are contemplated for use in connection with the various embodiments, i.e., anywhere that where animations can facilitate seamless context transitions. Accordingly, the below general purpose remote computer described below in Fig. 14 is but one example of a computing device.

[0084] Embodiments can partly be implemented via an operating system, for use
25 by a developer of services for a device or object, and/or included within application software that operates to perform one or more functional aspects of the various embodiments described herein. Software may be described in the general context of computer-executable instructions, such as program modules, being executed by one or
30 more computers, such as client workstations, servers or other devices. Those skilled in the art will appreciate that computer systems have a variety of configurations and protocols that can be used to communicate data, and thus, no particular configuration or protocol is considered limiting.

[0085] Fig. 14 thus illustrates an example of a suitable computing system environment 1400 in which one or aspects of the embodiments described herein can be implemented, although as made clear above, the computing system environment 1400 is only one example of a suitable computing environment and is not intended to suggest any limitation as to scope of use or functionality. In addition, the computing system environment 1400 is not intended to be interpreted as having any dependency relating to any one or combination of components illustrated in the exemplary computing system environment 1400.

[0086] With reference to Fig. 14, an exemplary remote device for implementing one or more embodiments includes a general purpose computing device in the form of a computer 1410. Components of computer 1410 may include, but are not limited to, a processing unit 1420, a system memory 1430, and a system bus 1422 that couples various system components including the system memory to the processing unit 1420.

[0087] Computer 1410 typically includes a variety of computer readable media and can be any available media that can be accessed by computer 1410. The system memory 1430 may include computer storage media in the form of volatile and/or nonvolatile memory such as read only memory (ROM) and/or random access memory (RAM). By way of example, and not limitation, system memory 1430 may also include an operating system, application programs, other program modules, and program data. According to a further example, computer 1410 can also include a variety of other media (not shown), which can include, without limitation, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disk (DVD) or other optical disk storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or other tangible and/or non-transitory media which can be used to store desired information.

[0088] A user can enter commands and information into the computer 1410 through input devices 1440. A monitor or other type of display device is also connected to the system bus 1422 via an interface, such as output interface 1450. In addition to a monitor, computers can also include other peripheral output devices such as speakers and a printer, which may be connected through output interface 1450.

[0089] The computer 1410 may operate in a networked or distributed environment using logical connections to one or more other remote computers, such as remote computer 1470. The remote computer 1470 may be a personal computer, a server, a router, a network PC, a peer device or other common network node, or any other remote

media consumption or transmission device, and may include any or all of the elements described above relative to the computer 1410. The logical connections depicted in Fig. 14 include a network 1472, such local area network (LAN) or a wide area network (WAN), but may also include other networks/buses. Such networking environments are
5 commonplace in homes, offices, enterprise-wide computer networks, intranets and the Internet.

[0090] As mentioned above, while exemplary embodiments have been described in connection with various computing devices and network architectures, the underlying concepts may be applied to any network system and any computing device or system in
10 which it is desirable to improve efficiency of resource usage.

[0091] Also, there are multiple ways to implement the same or similar functionality, e.g., an appropriate API, tool kit, driver code, operating system, control, standalone or downloadable software object, etc. which enables applications and services to take advantage of the techniques provided herein. Thus, embodiments herein are
15 contemplated from the standpoint of an API (or other software object), as well as from a software or hardware object that implements one or more embodiments as described herein. Thus, various embodiments described herein can have aspects that are wholly in hardware, partly in hardware and partly in software, as well as in software.

[0092] The word “exemplary” is used herein to mean serving as an example,
20 instance, or illustration. For the avoidance of doubt, the subject matter disclosed herein is not limited by such examples. In addition, any aspect or design described herein as “exemplary” is not necessarily to be construed as preferred or advantageous over other aspects or designs, nor is it meant to preclude equivalent exemplary structures and techniques known to those of ordinary skill in the art. Furthermore, to the extent that the
25 terms “includes,” “has,” “contains,” and other similar words are used, for the avoidance of doubt, such terms are intended to be inclusive in a manner similar to the term “comprising” as an open transition word without precluding any additional or other elements when employed in a claim.

[0093] As mentioned, the various techniques described herein may be
30 implemented in connection with hardware or software or, where appropriate, with a combination of both. As used herein, the terms “component,” “module,” “system” and the like are likewise intended to refer to a computer-related entity, either hardware, a combination of hardware and software, software, or software in execution. For example, a component may be, but is not limited to being, a process running on a processor, a

processor, an object, an executable, a thread of execution, a program, and/or a computer. By way of illustration, both an application running on computer and the computer can be a component. One or more components may reside within a process and/or thread of execution and a component may be localized on one computer and/or distributed between two or more computers.

[0094] The aforementioned systems have been described with respect to interaction between several components. It can be appreciated that such systems and components can include those components or specified sub-components, some of the specified components or sub-components, and/or additional components, and according to various permutations and combinations of the foregoing. Sub-components can also be implemented as components communicatively coupled to other components rather than included within parent components (hierarchical). Additionally, it can be noted that one or more components may be combined into a single component providing aggregate functionality or divided into several separate sub-components, and that any one or more middle layers, such as a coordination layer, may be provided to communicatively couple to such sub-components in order to provide integrated functionality. Any components described herein may also interact with one or more other components not specifically described herein but generally known by those of skill in the art.

[0095] In view of the exemplary systems described supra, methodologies that may be implemented in accordance with the described subject matter can also be appreciated with reference to the flowcharts of the various figures. While for purposes of simplicity of explanation, the methodologies are shown and described as a series of blocks, it is to be understood and appreciated that the various embodiments are not limited by the order of the blocks, as some blocks may occur in different orders and/or concurrently with other blocks from what is depicted and described herein. Where non-sequential, or branched, flow is illustrated via flowchart, it can be appreciated that various other branches, flow paths, and orders of the blocks, may be implemented which achieve the same or a similar result. Moreover, some illustrated blocks are optional in implementing the methodologies described hereinafter.

[0096] In addition to the various embodiments described herein, it is to be understood that other similar embodiments can be used or modifications and additions can be made to the described embodiment(s) for performing the same or equivalent function of the corresponding embodiment(s) without deviating therefrom. Still further, multiple processing chips or multiple devices can share the performance of one or more functions

described herein, and similarly, storage can be effected across a plurality of devices. Accordingly, the invention is not to be limited to any single embodiment, but rather is to be construed in breadth, spirit and scope in accordance with the appended claims.

CLAIMS

What is claimed is:

1. A method facilitated by at least one processor of a computing device, comprising:

5 receiving initial information from an initiator application, wherein the initial information relates to a transitional animation that visualizes a shift in context of the computing device from the initiator application to a destination application;

receiving final information, associated with the transitional animation, from the destination application;

10 generating the transitional animation based at least in part on the initial information and the final information; and

rendering the transitional animation on a display device of the computing device.

2. The method of claim 1, further comprising:

generating an animation identifier that corresponds to the transitional animation;

15 and

issuing the animation identifier to at least one of the initiator application or the destination application.

3. The method of claim 1, wherein the generating the transitional animation further comprises creating a plurality of intermediate graphical representations that bridge a
20 difference between an initial graphical representation included in the initial information and a final graphical representation specified in the final information,

wherein a number of the plurality of intermediate graphical representations created is in accordance with a predetermined frame rate of the transitional animation.

4. The method of claim 3, further comprising determining a set of position

25 information associated with the plurality of intermediate graphical representations including determining the position information based on at least one of initial position information in the initial information, final position information included in the final information, or other position information included in the set.

5. The method of claim 4, wherein the rendering the transitional animation further
30 comprises:

ordering the initial graphical representation, the plurality of intermediate graphical representations, and the final graphical representation to create a frame order; and

sequentially and individually rendering the initial graphical representation, the plurality of intermediate graphical representations, and the final graphical representation in accordance with the frame order, and

determining a time between renderings based on the predetermined frame rate.

5 6. The method of claim 1, wherein the rendering the transitional animation further comprises enforcing a z-position of the transitional animation wherein the transitional animation remains a top-most object on the display device.

7. The method of claim 1, further comprising waiting for a readiness indicator from the destination application before commencing the rendering of the transitional animation.

10 8. A system, comprising:

an animation coordination component, including:

an input component configured to receive initial information from one or more initiator applications and to receive final information from one or more destination applications; and

15 an animation component configured to create and to render animations on a display device, wherein the animations provide seamless transitions from environments associated with the one or more initiator applications to environments associated with the one or more destination applications.

9. The system of claim 8, wherein the animation coordination component further
20 comprises an identifier generation component configured to issue animation identifiers to individual animations coordinated by the animation coordination component.

10. The system of claim 9, wherein the animation coordination component further comprises an output component configured to transmit the animation identifiers to the one or more initiator applications as a receipt of the initial information.

25 11. The system of claim 8, wherein the animation coordination component further comprises a frame generation component configured to create animation frames for the animations to be drawn by the animation component.

12. The system of claim 8, wherein the initial information includes at least one of
30 of the objects in the one or more initiator applications, respective positions of the initial graphical representations, or respective sizes of the initial graphical representations.

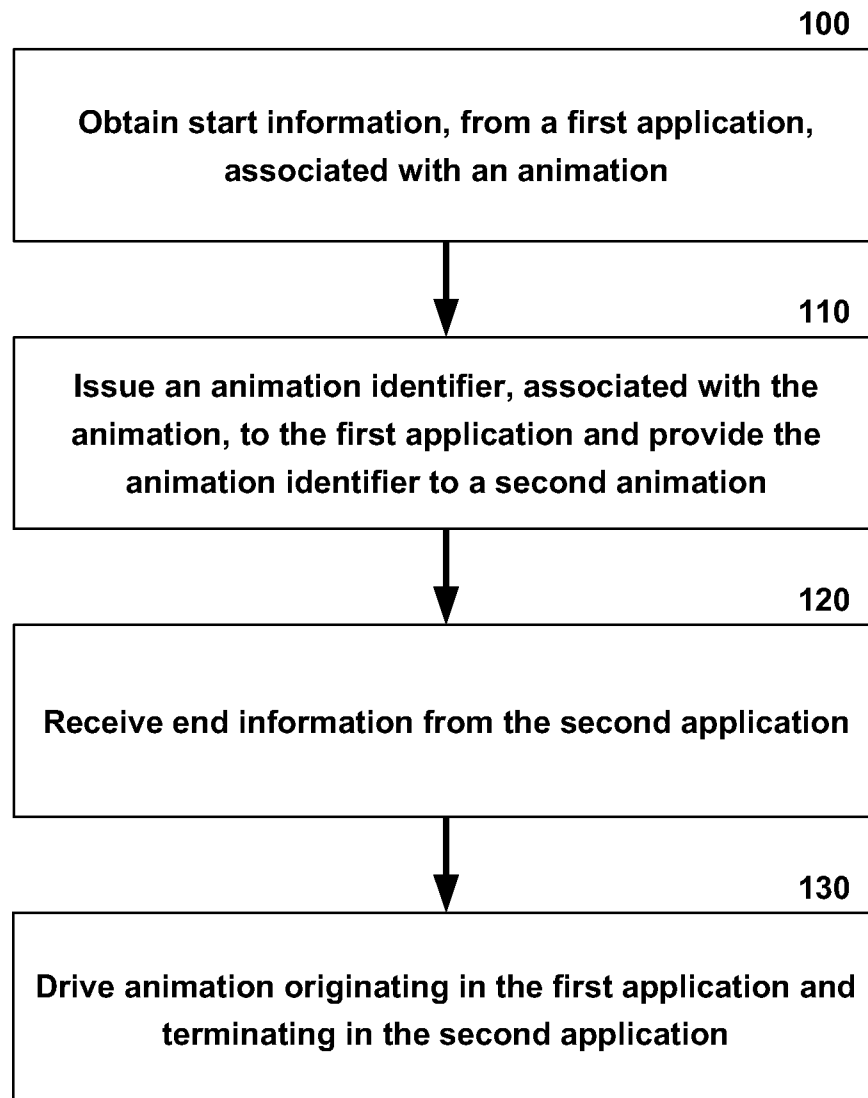
13. The system of claim 8, wherein the final information includes at least one of final graphical representations of objects in the one or more destination applications or types of the objects in the one or more destination applications.

14. The system of claim 8, wherein the final information includes at least one of
5 respective positions of the final graphical representations, or respective sizes of the final graphical representations.

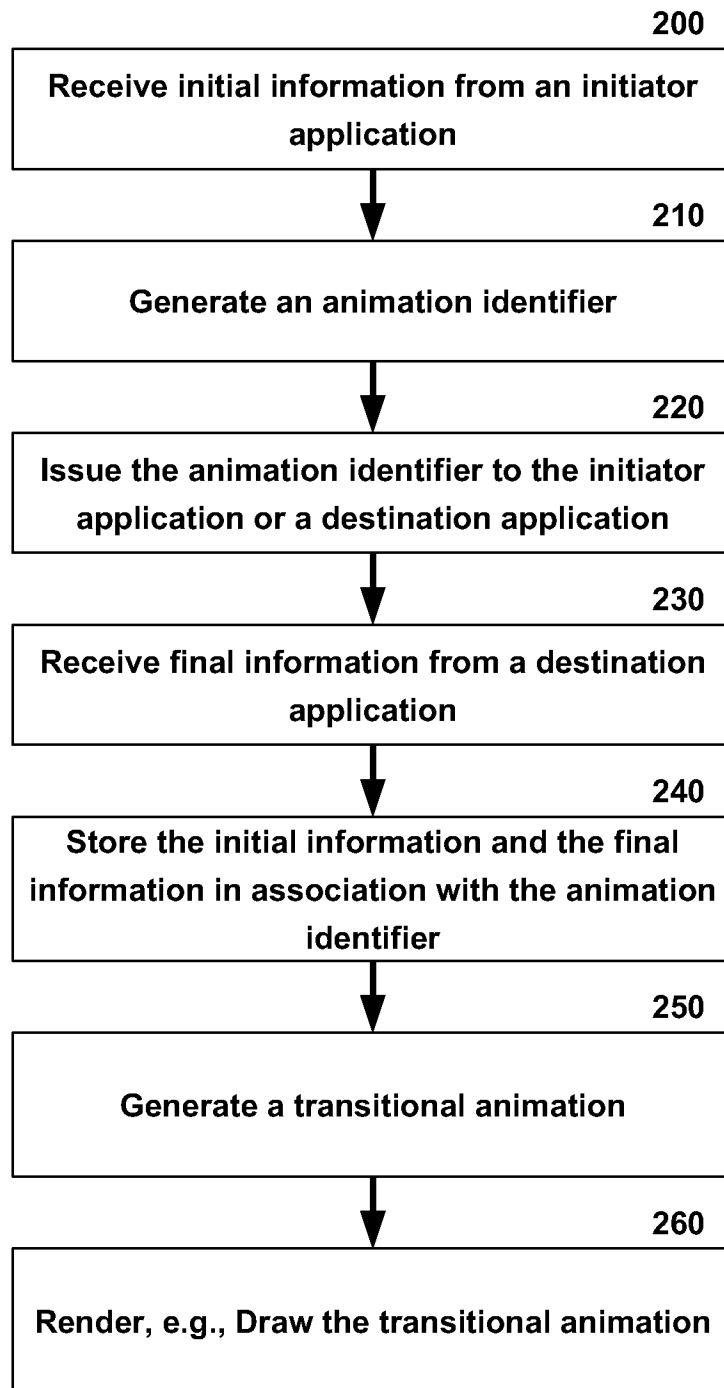
15. The system of claim 8, wherein the animation coordination component is integrated into an operating system.

10

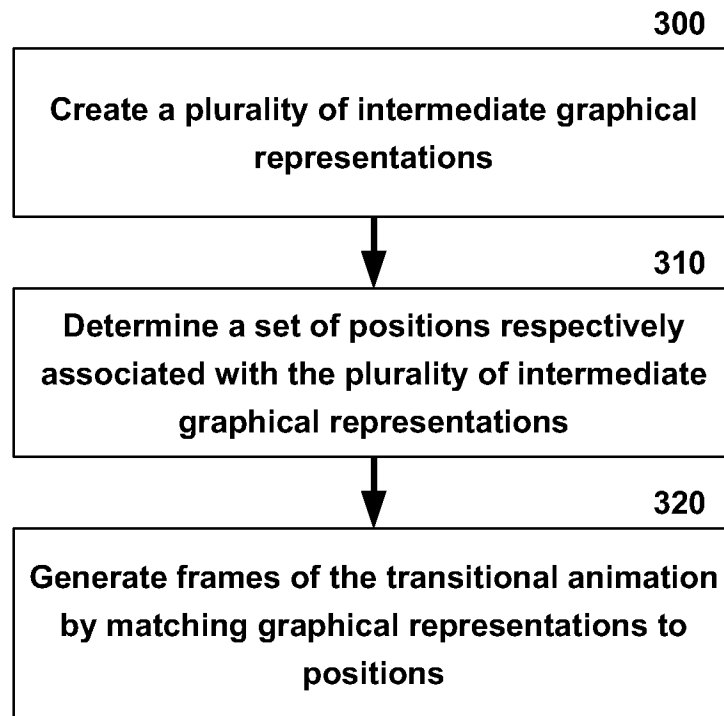
1/14

**FIG. 1**

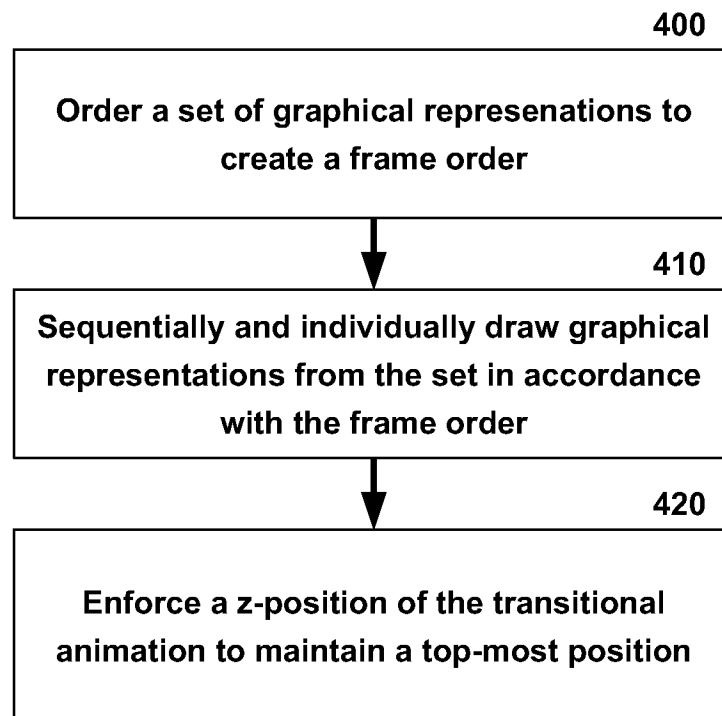
2/14

**FIG. 2**

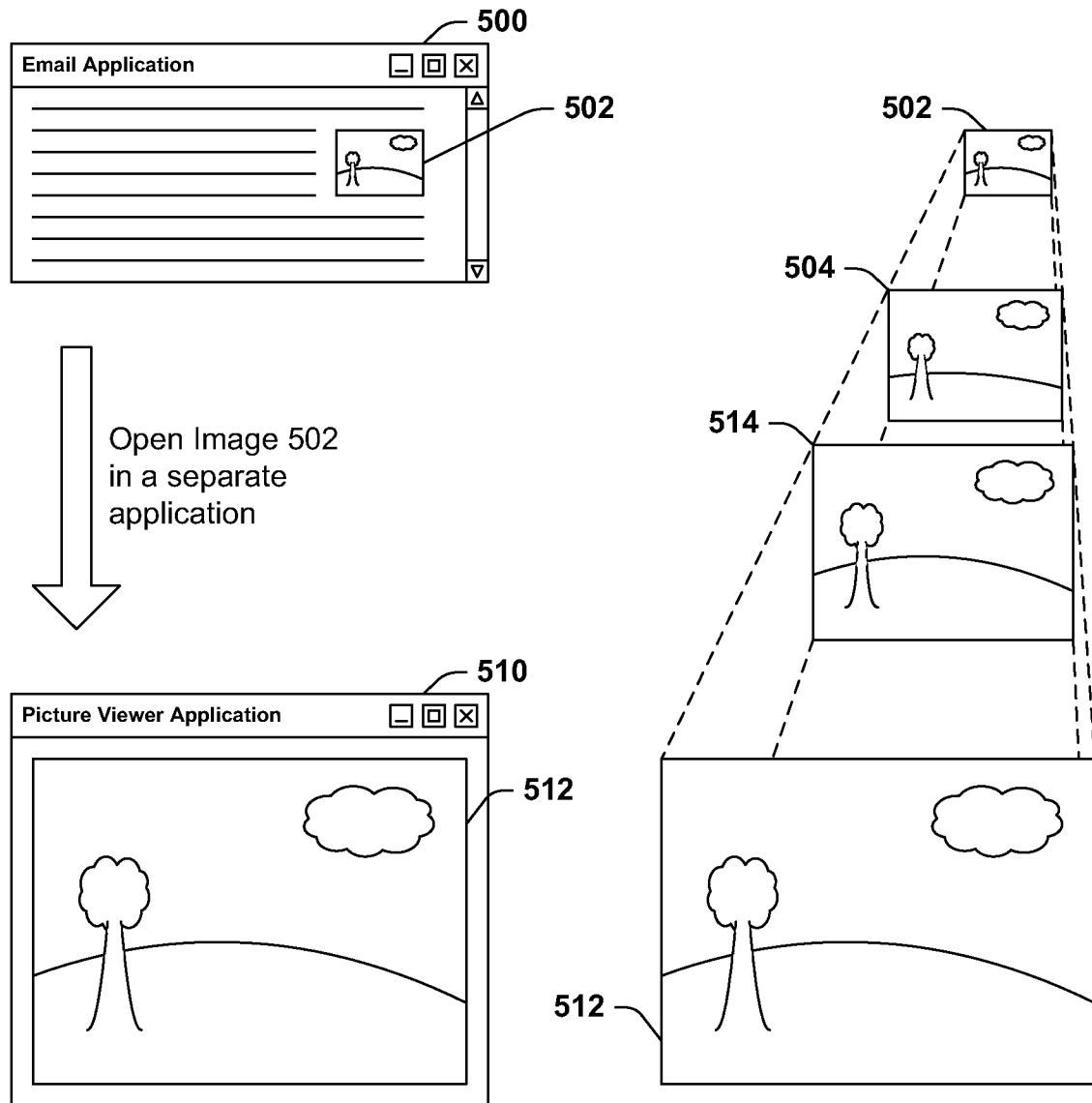
3/14

**FIG. 3**

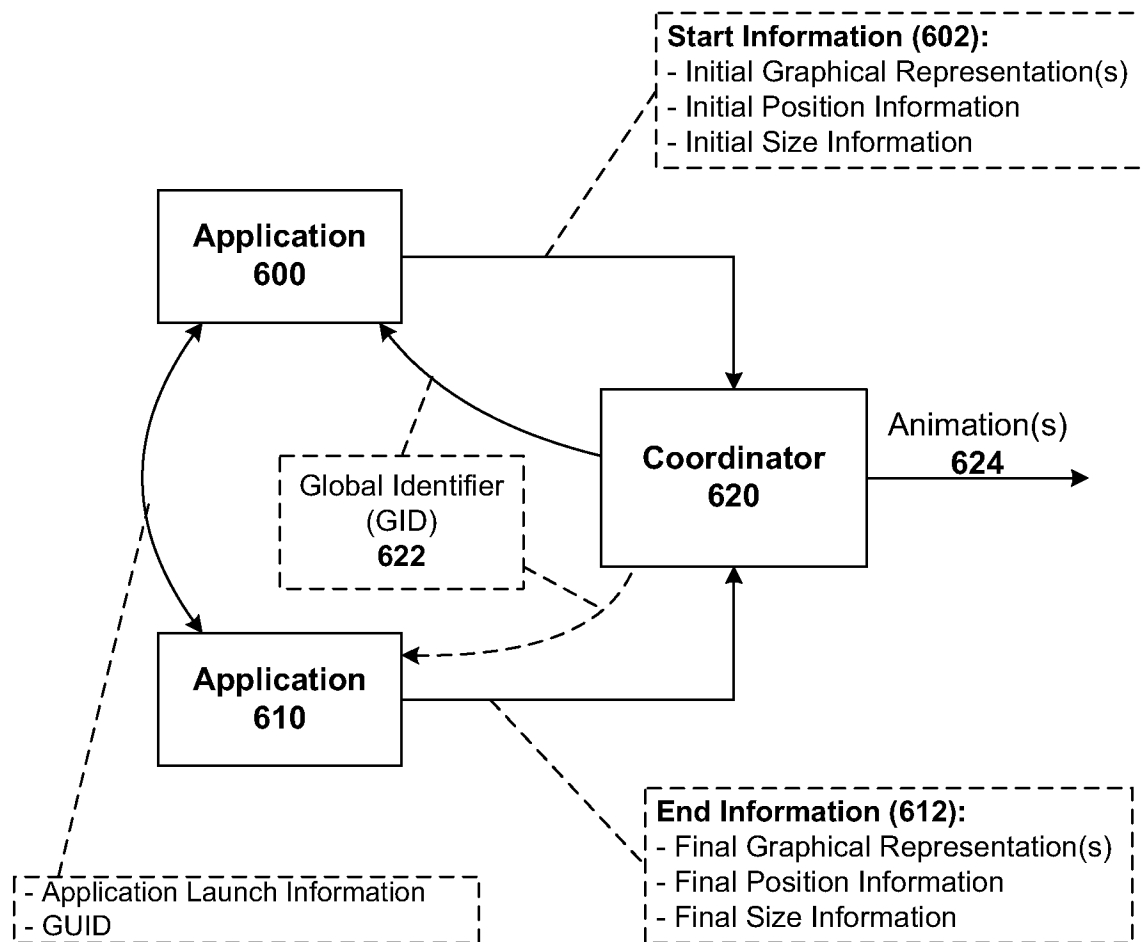
4/14

**FIG. 4**

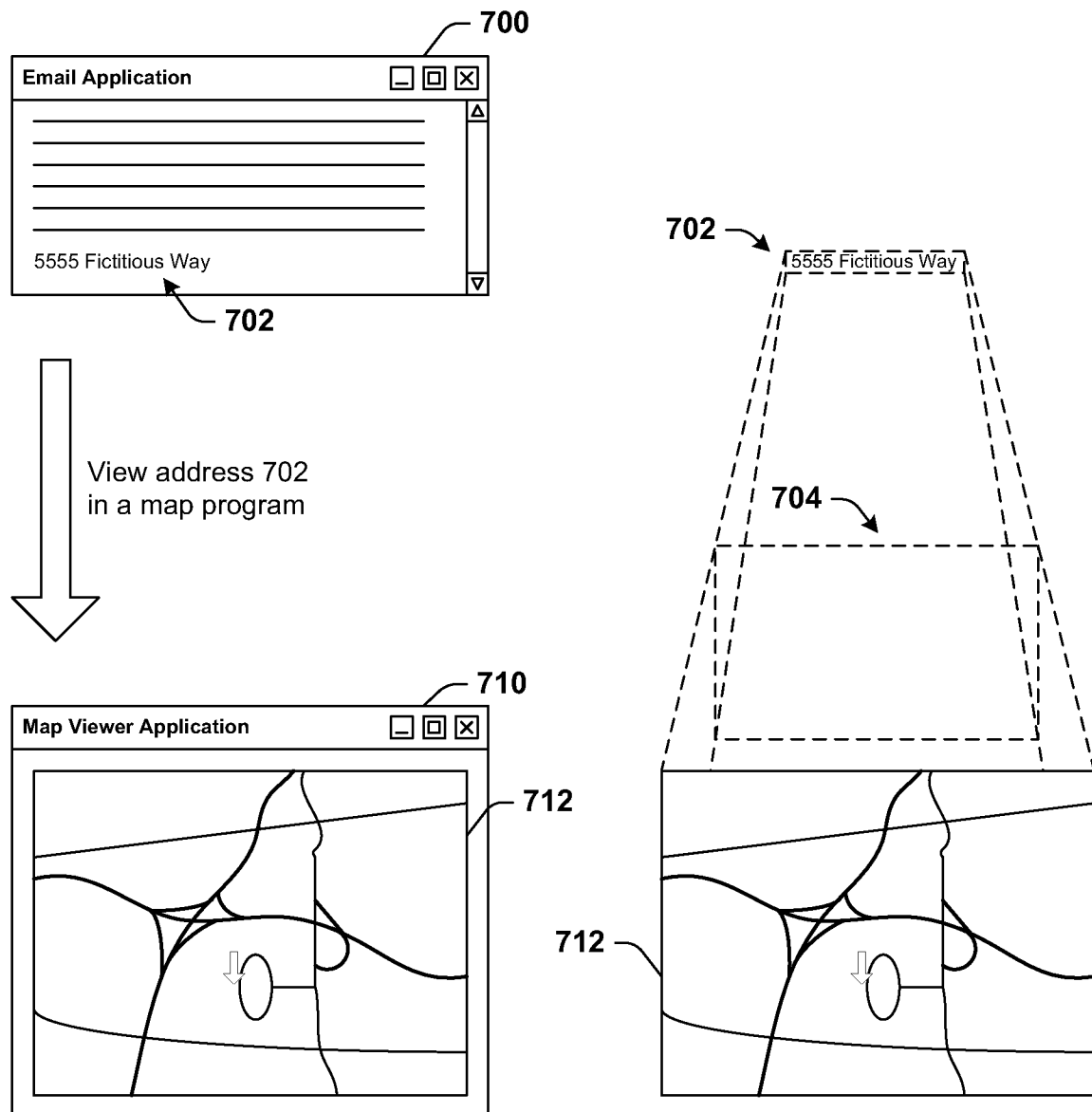
5/14

**FIG. 5**

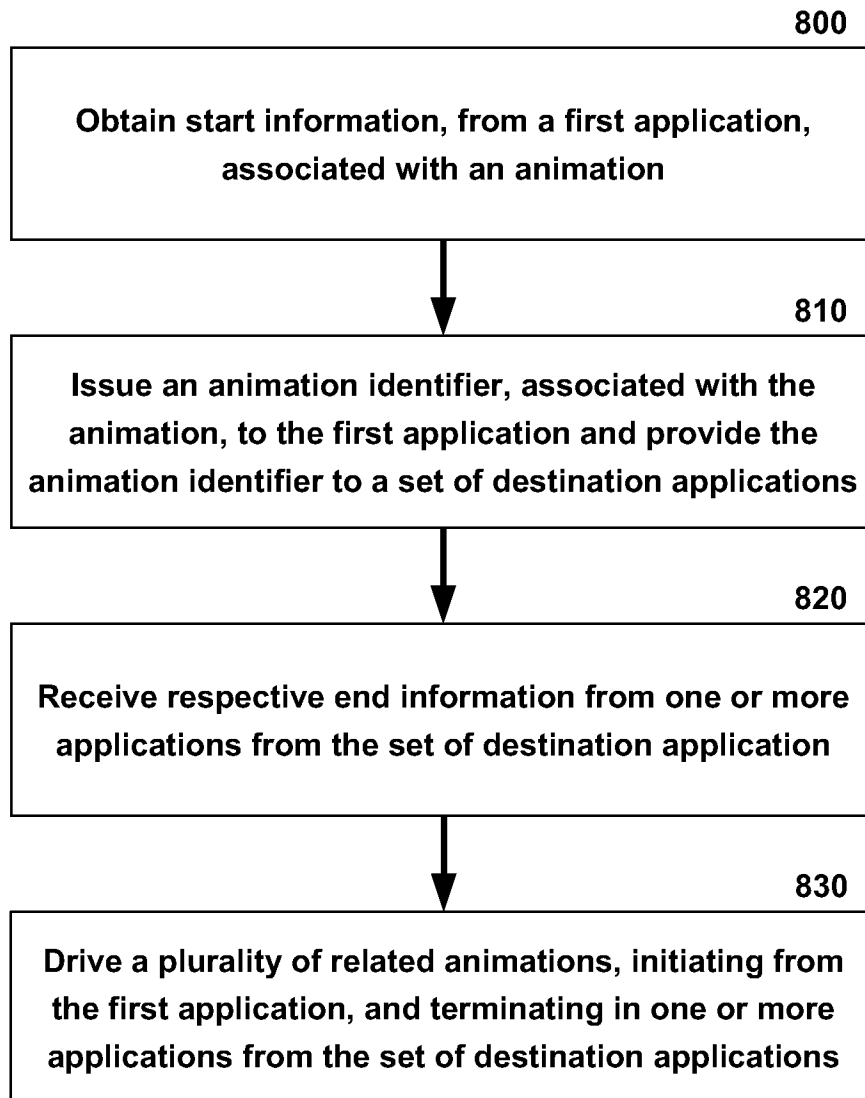
6/14

**FIG. 6**

7/14

**FIG. 7**

8/14

**FIG. 8**

9/14

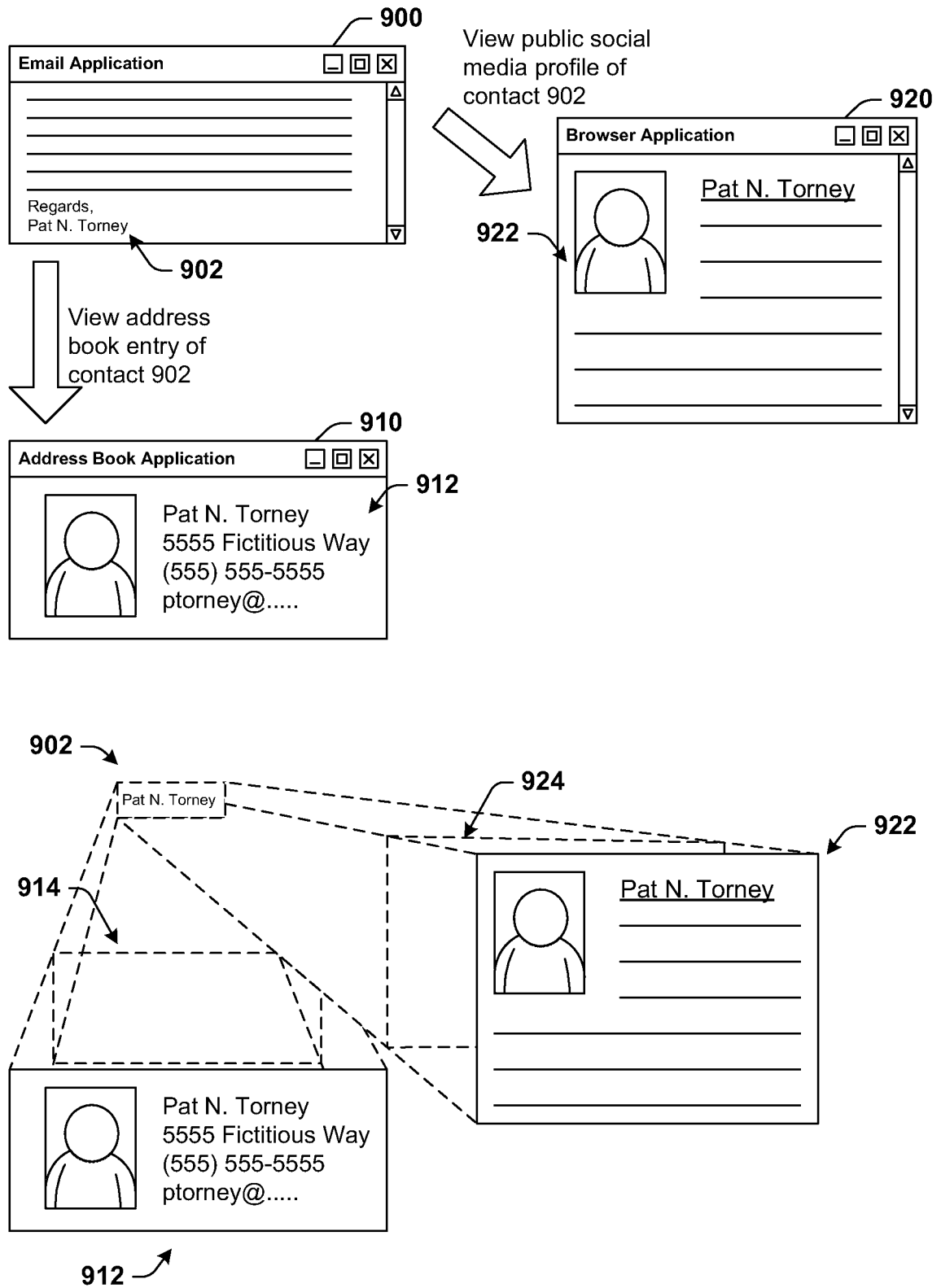
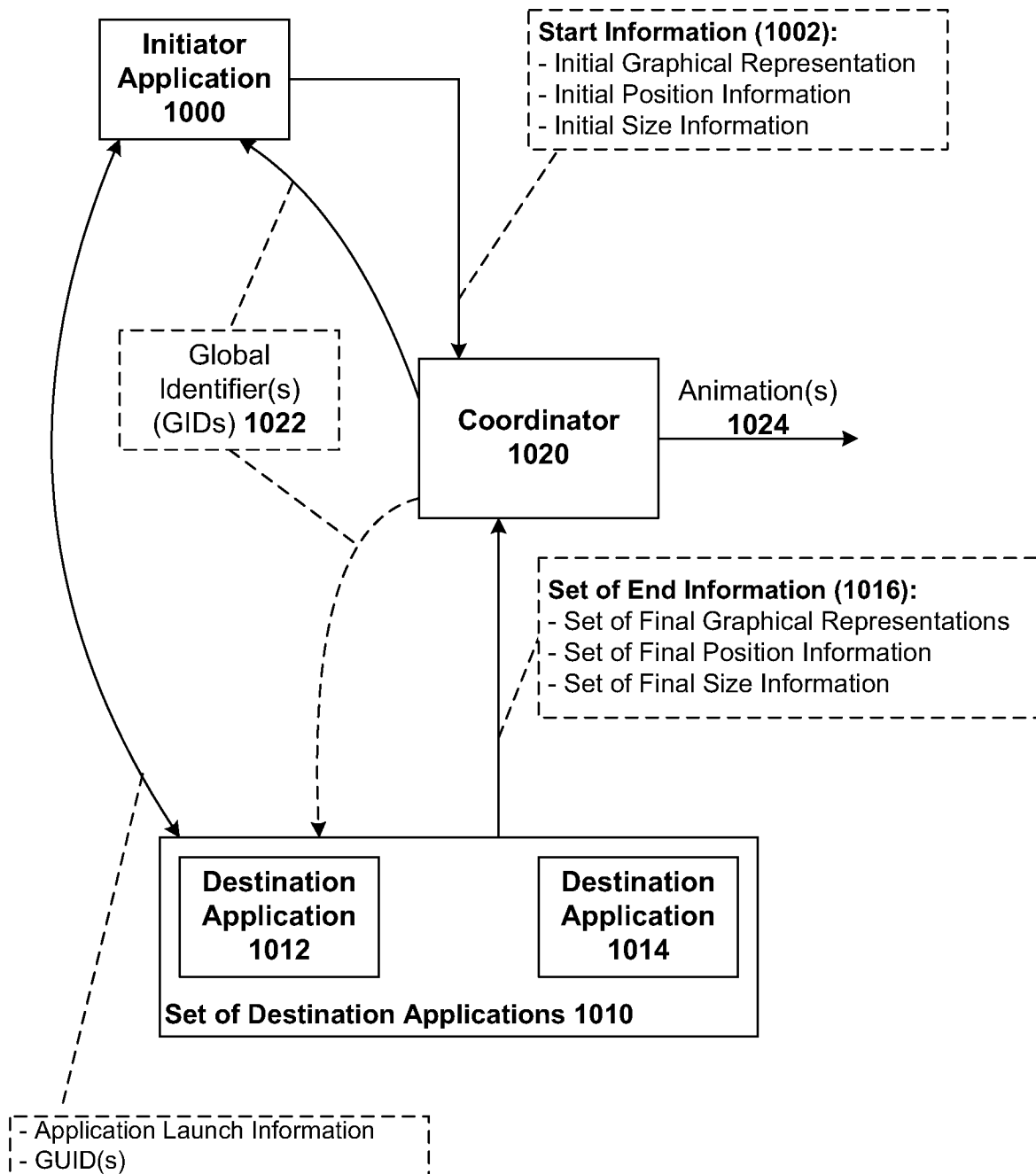
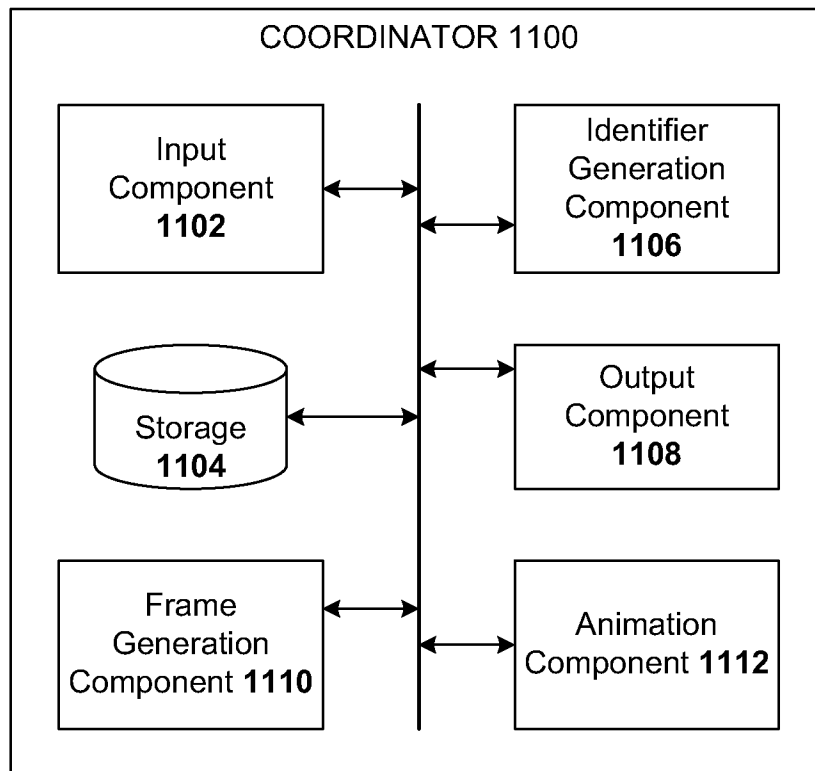


FIG. 9

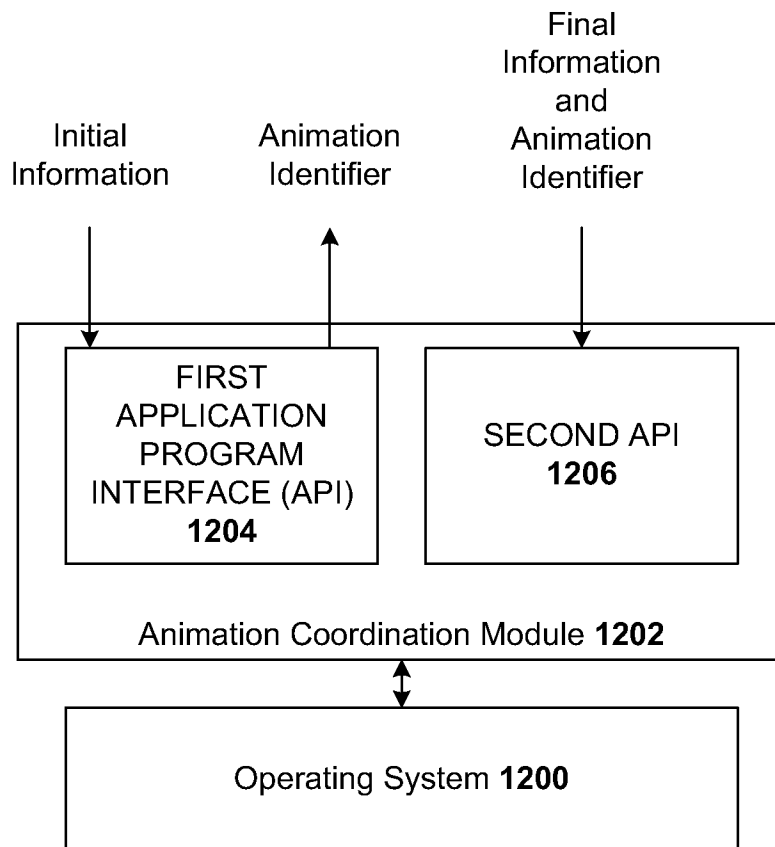
10/14

**FIG. 10**

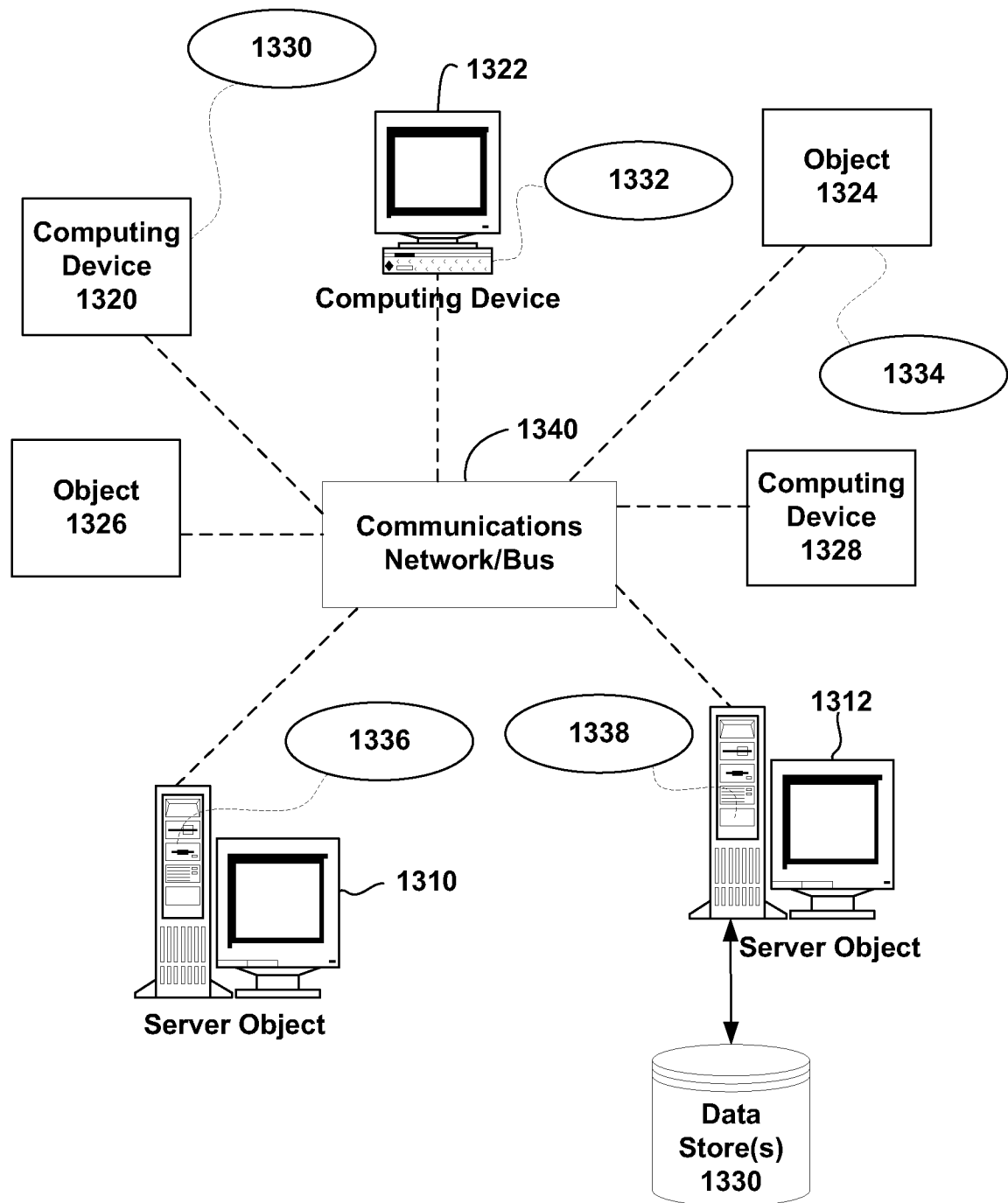
11/14

**FIG. 11**

12/14

**FIG. 12**

13/14

**FIG. 13**

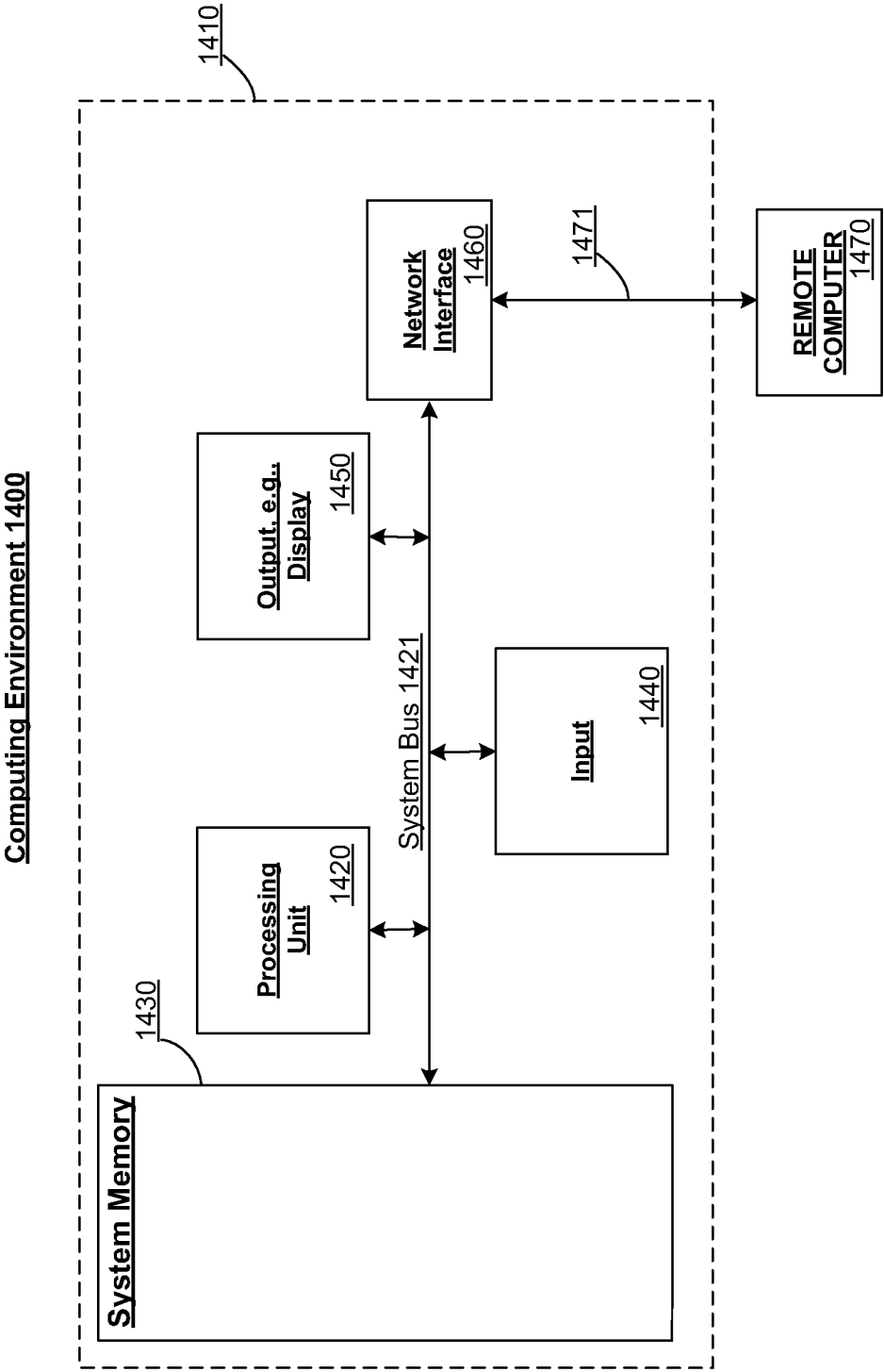


FIG. 14

A. CLASSIFICATION OF SUBJECT MATTER**G06T 13/00(2011.01)i, G06F 9/44(2006.01)i**

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06T 13/00; G06F 11/30; G06T 11/00; G06F 9/46; G09G 5/00

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Korean utility models and applications for utility models

Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

eKOMPASS(KIPO internal) & Keywords: context, inter-application, transition, animation

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	WO 2005-103900 A1 (GOOGLE, INC. et al.) 03 November 2005 See paragraph [0076] - paragraph [0084]; claim 30 and figure 5.	1-15
Y	KIM, JI YONG. A Study of Classification and Application for Transition Effect in Graphic User Interface. Master Thesis. Kookmin University, Korea, 28 February 2011, pp.32-48. See page 32, line 1 - page 48, line 21.	1-15
A	WO 2006-017388 A1 (SOFTRICITY, INC. et al.) 16 February 2006 See paragraph [0059] - paragraph [0117]; claims 1-8 and figure 5.	1-15
A	US 2008-0165207 A1 (KOSHAKE RICHARD L. et al.) 10 July 2008 See paragraph [0037] - paragraph [0046]; claim 1 and figure 2.	1-15



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

29 MAY 2012 (29.05.2012)

Date of mailing of the international search report

29 MAY 2012 (29.05.2012)

Name and mailing address of the ISA/KR

Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan
City, 302-701, Republic of Korea

Facsimile No. 82-42-472-7140

Authorized officer

Lee Jumi

Telephone No. 82-42-481-8529



INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2011/063458

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 2005-103900 A1	03.11.2005	CA 2560747 A1	03.11.2005
		CN 101111836 A0	23.01.2008
		CN 1934539 A	21.03.2007
		CN 1934539 C0	22.10.2008
		EP 1745360 A2	24.01.2007
		EP 1745360 B1	11.05.2011
		EP 1745379 A1	24.01.2007
		JP 04-812747 B2	02.09.2011
		JP 2007-534057 A	22.11.2007
		KR 10-0952379 B1	14.04.2010
		KR20060132946A	22.12.2006
		US 2005-0234848 A1	20.10.2005
		US 2005-0246588 A1	03.11.2005
		US 7680809 B2	16.03.2010
		US 7725508 B2	25.05.2010
		US 7941439 B1	10.05.2011
		WO 2005-098594 A2	20.10.2005
		WO 2005-098594 A3	20.10.2005
		WO 2005-103900 A8	09.10.2008
WO 2006-017388 A1	16.02.2006	CN 100481013 C	22.04.2009
		CN 101048735 A0	03.10.2007
		EP 1784725 A1	16.05.2007
		JP 2008-509475 A	27.03.2008
		JP 2008-509475 T	27.03.2008
		KR 10-2007-0057783 A	07.06.2007
		KR20070057783A	07.06.2007
		US 2006-0036570 A1	16.02.2006
		US 7962918 B2	14.06.2011
US 2008-0165207 A1	10.07.2008	AU 2007-342198 A1	17.07.2008
		CA 2667589 A1	17.07.2008
		EP 2102821 A2	23.09.2009
		JP 2010-515968 A	13.05.2010
		JP 2010-515968 T	13.05.2010
		KR 10-2009-0096698 A	14.09.2009
		KR20090096698A	14.09.2009
		WO 2008-085677 A2	17.07.2008
		WO 2008-085677 A3	25.09.2008
		WO 2008-085677 A3	17.07.2008