(19) OFICIUL DE STAT
PENTRU INVENŢII ŞI MĂRCI
Bucureşti

ROMÂNIA

(11) **RO 128172 A2**
(51) Int.Cl.
G06T 13/00 (2006.01),
H04N 7/24 (2006.01)

(12) # CERERE DE BREVET DE INVENŢIE

(21) Nr. cerere: **a 2011 00322**

(22) Data de depozit: **07.04.2011**

(41) Data publicării cererii:
28.02.2013 BOPI nr. **2/2013**

(71) Solicitant:
• **CHICULIŢĂ ALEXANDRU**,
STR. TRESTIANA NR. 3, BL. 8B, SC. A, ET. 6, AP. 27, SECTOR 4, BUCUREŞTI, B, RO

(72) Inventatori:
• **CHICULIŢĂ ALEXANDRU**,
STR. TRESTIANA NR. 3, BL. 8B, SC. A, ET. 6, AP. 27, SECTOR 4, BUCUREŞTI, B, RO

(74) Mandatar:
CABINET ENPORA S.R.L.,
STR. GEORGE CĂLINESCU NR. 52A, AP. 1, SECTOR 1, BUCUREŞTI

(54) # METODE ŞI SISTEME PENTRU REPREZENTAREA ANIMAŢIEI COMPLEXE PRIN UTILIZAREA POSIBILITĂŢILOR DE SCRIPTING ALE APLICAŢIILOR DE RENDERING

(57) **Rezumat:**

Invenţia se referă la o metodă, un dispozitiv şi un produs program de calculator pentru reprezentarea animaţiei complexe. Metoda conform invenţiei constă din accesarea datelor care definesc o secvenţă de animaţie înfăţişând mişcarea în timp cel puţin a unui obiect, accesarea primitivelor de animaţie care identifică datele suportate de un limbaj de marcare, analizarea datelor care definesc secvenţa de animaţie, pentru a determina o primă porţiune a secvenţei de animaţie care poate să fie reprezentată prin intermediul utilizării unui set de resurse vizuale animate, prin utilizarea primitivelor de animaţie, şi o a doua porţiune a secvenţei de animaţie care poate să fie reprezentată prin utilizarea unui limbaj de scriptare, pentru a controla tranziţiile dintre cadrele care alcătuiesc a doua porţiune, şi generarea unui pachet având în componenţă un cod de marcare ce face referire la resursele vizuale şi care defineşte un aspect al cadrelor, o foaie de stil ce face referire la primitivele de animaţie, şi un obiect de date structurate, codul de marcare fiind generat pentru a determina o aplicaţie de randare să furnizeze secvenţa de animaţie prin utilizarea limbajului de scriptare pentru a coordona randarea fiecăreia dintre prima porţiune şi a doua porţiune, în conformitate cu un parametru inclus în obiectul de date structurate, prima porţiune fiind randată prin aplicarea primitivelor de animaţie ca stiluri la setul de resurse vizuale, şi a doua porţiune fiind randată prin extragerea mulţimii de cadre şi tranziţionarea între cadre folosind limbajul de scriptare. Dispozitivul computerizat, conform invenţiei, are în componenţă un interconector de hardware şi un element de hardware pentru prelucrarea datelor, ce pune în aplicare un motor pentru codarea animaţiei şi pentru analiza datelor de cronologie care definesc o secvenţă de animaţie, şi generează un pachet de cod reprezentând secvenţa de animaţie şi având în componenţă un cod de marcare şi un obiect de date structurate.

Revendicări: 20
Figuri: 6

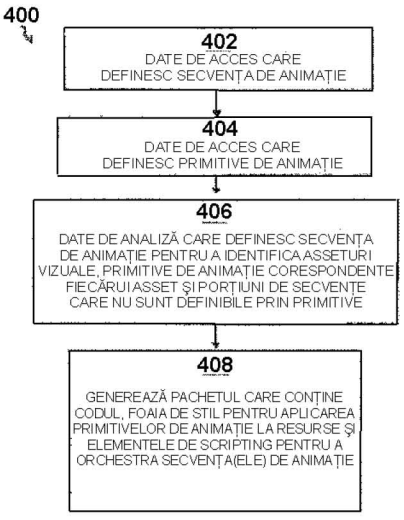

Fig. 4

RO 128172 A2

Stadiul tehnicii mondiale în domeniul invenției

[0001] Animația este o o modalitate răspândită pentru a furniza conținut, fie pentru scopuri artistice, fie de reclamă sau pentru alte scopuri. Aplicațiile de runtime, cum sunt cele executate folosind Adobe® Flash® player (disponibil de la Adobe Systems Incorporated of San Jose, California), sunt o opțiune eficientă pentru distribuirea de conținut animat prin intermediul Internetului. De exemplu, o aplicație de runtime poate avea în componență un cod care, atunci când este executat într-un mediu de runtime corespondent, prezintă o secvență de animație dorită, cum ar fi unul sau mai multe obiecte animate care se mişcă şi/sau se modifică altfel într-o modalitate care variază cu timpul pe un stage într-o interfață renderată de către aplicație de runtime. Cu toate acestea, animația pe bază de runtime poate să nu fie disponibilă întotdeauna – de exemplu, anumite dispozitive sau platforme pot să nu suporte utilizarea unui mediu de runtime. Dezvoltatorii pot cu toate acestea dori să furnizeze conținut animat pentru astfel de platforme, cum ar fi conținut animat pentru utilizarea cu aplicații de rendering (de exemplu, browsere) care pot să furnizeze animație dar nu pot să lucreze cu mediul de runtime.

Rezumat

[0002] Un dispozitiv computerizat include un interconector de hardware şi un element de hardware pentru procesarea datelor (de exemplu, un procesor şi/sau logică de hardware) interfațată către interconectorul de hardware. Dispozitivul computerizat implementează un motor de codare a animației pentru a analiza datele de timeline care definesc o secvență de animație şi generează un pachet de cod. Pachetul cod poate să reprezinte secvența de animație care utilizează codul de markup care defineşte aspectul renderat al mai multor cadre şi un obiect de date structurat aflat de asemenea în componența pachetului cod şi definind un parametru utilizat de către limbajul de scripting în tranziția între cadre. Codul de markup poate de asemenea să aibă în componență o referință la o resursă vizuală inclusă într-un cadru. Pachetul cod are de asemenea în componență o foaie de stil în cascadă (Cascading style sheet) care defineşte un primitiv de animație ca pe un stil care trebuie să fie aplicat asset-ului pentru a reproduce una sau mai multe porțiuni din secvența de animație fără a tranziționa între cadre.

[0003] Aceste modalităţi de realizare ilustrative sunt discutate nu pentru a limita prezentul subiect, ci pentru a pune la dispoziţie o scurtă introducere. Modalităţi de realizare adiţionale includ medii citibile pe calculator care încorporează o aplicaţie configurată în conformitate cu asapecte ale prezentului subiect pentru a pune la dispoziţie un motor pentru codarea animaţiei. Modalităţile de realizare includ de asemenea metode puse în aplicare pe calculator pentru generarea pachetelor de cod care pot să fie procesate de către o aplicaţie de rendering pentru a furniza animaţie bazată pe cadre de rendering şi pentru a invoca primitive de animaţie native pentru aplicaţia de rendering. Acestea şi alte modalităţi de realizare sunt descrise în cele ce urmează în cadrul Descrierii detaliate. Obiectele şi avantajele prezentului subiect pot să fie determinate după trecerea în revistă a specificaţiei şi/sau punerea în practică a unei modalităţi de realizare configurată în conformitate cu unul sau mai multe din aspectele dezvăluite aici.

Scurtă descriere a desenelor

[0004] O completă şi accesibilă expunere este detaliată mai amănunţit în ceea ce a rămas din specificaţia brevetului. Specificaţia face referire la următoarele figuri care îi sunt anexate.

[0005] Figura 1 este o diagramă a unei secvenţe de animaţie ilustrative.

[0006] Figura 2 este o diagramă care prezintă un dispozitiv de calcul ilustrativ care pune în aplicare un motor pentru codarea animaţiei.

[0007] Figura 3 este o diagramă de flux care prezintă procesul de generare a codului.

[0008] Figura 8 este o schemă logică de program care prezintă o metodă ilustrativă pentru generarea unui cod pe baza datelor de timeline.

[0009] Figura 5 este o schemă logică de program care prezintă un exemplu al analizării datelor de timeline şi generării unui pachet de cod.

[0010] Figura 6 este o schemă bloc care prezintă o arhitectură ilustrativă pentru un mediu de dezvoltare care utilizează un motor pentru codarea animației configurat în conformitate cu prezentul subiect.

Descrierea detaliată

[0011] Modalitățile de realizare expuse în cele ce urmează includ sisteme de calcul, metode şi medii citibile pe calculator care încorporează cod. Figura 1, de exemplu, prezintă o secvență de animație 102 ilustrativă care reprezintă o schimbare în aspectul în timp al unui obiect 103. Secvența de animație 102 este analizată prin intermediul unui motor de codare pentru animație 104, care accesează datele de timeline 106 care reprezintă secvența de animație şi produce un pachet de cod 108 care reprezintă secvența de animație utilizând un cod de markup, o foaie de stil şi un obiect de date structurate (de exemplu, un JSON (JavaScript Object Notation) care controlează modul în care un limbaj de scripting (de exemplu, JavaScript) poate să fie utilizat pentru a controla tranzițiile dintre porțiunile de animație după cum sunt reprezentate în pachetul de cod (adică, scriptingul este utilizat pentru a orchestra animația).

[0012] În cadrul acestui exemplu particular, secvența de animație 102 înfățişează o minge 103 care trece în lungul ecranului după cum este arătat la 103A, se deformează sare de la solul G după cum este arătat la 103B şi îşi schimbă culoarea după săritură şi continuă după cum este prezentat la 103C. Secvența de animație 102 poate, de exemplu, să fie reprezentată utilizând datele de timeline 106 incluse într-un fişier sursă sau compilat pentru executarea unui mediu de runtime, cum ar fi o sursă FLA care poate să fie compilată într-un fişier SWF executabil utilizând mediile de runtime Adobe® Flash® sau AIR® (ambele disponibile de la Adobe Systems Incorporated of San Jose, California). Datele de timeline 106 pot să fie într-o altă formă compilată sau necompilată.

[0013] Nu toate platformele de calcul pot să suporte redarea folosind un mediu de runtime. Astfel, motorul pentru codarea animației 104 poate să fie utilizat pentru a transforma datele de timeline 106 în pachetul de cod 108, care, atunci când este renderat prin intermediul unei aplicații de rendering (de exemplu, un browser)

asigură aceeaşi sau aproape aceeaşi secvenţă de animaţie 102. De exemplu, pachetul de cod 108 poate avea în componenţă codul de markup 110, cum este codul HTML, care defineşte un aspect renderat al tuturor cadrelor de animaţie după cum este prezentat la 112A, 112B şi 112C. Unele dintre cadre sau toate cadrele 112 pot să includă codul de asset 114/115 pentru a extrage sau a rendera un asset vizual (de exemplu, o grafică vector sau raster a mingii 103) din cadru. Foaia de stil 116 poate să fie inclusă în codul de markup 110 sau într-un fişier separat la care face referire codul de markup 110 şi care poate să fie folosit pentru a defini un stil, care, atunci când este aplicat unuia sau mai multor asset-uri vizuale, aplică un primitiv de animaţie pentrru a schimba aspectul asset-ului vizual.

[0014] Obiectul de date structurate 118 poate să includă unul sau mai mulţi parametri care să identifice diverse componente ale pachetului de cod 108 şi informaţia de timing pentru utilizarea de către un limbaj de scripting pentru a controla, atunci când cadrele renderate sunt vizibile, tranziţia dintre cadrele 112 şi pentru a controla atunci când stilurile definite în foaia de stil 116 sunt utilizate pentru a aplica primitive de animaţie.

[0015] De exemplu, datele de timeline 106 pot să includă o referinţă la grafica raster sau la grafica vetorială care defineşte un aspect al mingii 103, zona solului G şi poziţiile mingii 103 şi ale zonei de sol G într-un număr de cadre cheie (suprafaţa solului G având probabil o singură poziţie). Dat fiind faptul că datele de timeline 106 sunt într-un format care trebuie compilat sau renderat de către un mediu de runtime, datele de timeline 106 pot să se bazeze pe mediul de runtime pentru a folosi tranziţiile vizuale prezentate la 103A, 103C şi chiar deformaţia prezentată la 103B. Pe de altă parte, pachetul de cod 108 poate să fie destinat pentru o aplicaţie de rendering (de exemplu, un browser) care are capacităţi de animaţie native limitate. În consecinţă, motorul pentru codarea animaţiei 104 poate să genereze cod de markup pentru a defini porţiuni ale secvenţei de animaţie 102 în conformitate cu capacităţile aşteptate ale aplicaţiei de rendering.

[0016] De exemplu, deplasarea mingii 103 după cum este prezentată la 103A poate să corespundă unui primitiv de animaţie suportat de către un browser. În felul acesta, motorul pentru codarea animaţiei 104 poate să definească un cadru renderat

120A al cărui aspect a fost reprezentat folosind codul 112A potrivit (de exemplu, un element <canvas> HTML) cu un asset vizual 114 (de exemplu, un SVG (grafică vectorială proporţională) sau un alt fişier de imagine care reprezintă mingea 103) în cadru. Un stil poate să fie inclus în foaia de stil 103 de la poziţia sa de început până la poziţia sa de sfârşit înaintede porţiunea de săritură 103B.

[0017] Aplicaţia de rendering poate să nu suporte nativ un primitiv de animaţie care corespunde cu deformarea mingii 103 după cum este prezentat la 103B şi în felul acesta motorul pentru codarea animaţiei 104 poate să includă codul de markup 112B pentru a defini mai multe dintre cadrele renderate 120B fiecare din acestea reprezentând un aspect al porţiunii 103B a secvenţei de animaţie la o anumită etapă din timp. De exemplu, codul de markup 112B poate să aibă în componenţă un element <canvas> cu comenzi corespunzătoare pentru a reproduce vizual fiecare cadru renderat.

[0018] Apoi, poate să fie definit în codul 112C un alt cadru renderat 120C, cu un asset 115 care reprezintă mingea 103 cu culoarea schimbată, cu un alt primitiv de animaţie utilizat pentru a reproduce mişcarea prezentată la 103C. Obiectul de date structurate 118 poate să fie populat cu date care identifică diversele cadre şi parametri corespunzători astfel încât JavaScript sau un alt limbaj de scripting poate să fie folosit pentru a orchestra animaţia. Aceasta înseamnă că scripting-ul poate să fie folosit pentru a afişa cadrele 120A-120B-120C în ordine prin controlul vizibilităţii acelor cadre şi al tranziţiei dintre cadre. Suplimentar, scriptingul poate să fie folosit pentru a activa primitivele de animaţie în timp ce este afişat un cadru, ca corespunzător (de exemplu, pentru a aplica stiluri din foaia de stil 116 pentru a deplasa asset-ul 114, 115 care reprezintă mingea 103).

[0019] Acum ne vom referi mai în detaliu la diverse şi alternative modalităţi de realizare cu titlu de exemplificări cât şi la desenele care le însoţesc. Fiecare exemplu este pus la dispoziţie în scopul explicării şi nu al limitării. Va fi evident pentru specialiştii din domeniu că pot să fie aduse modificări şi variaţiuni. De exemplu, caracteristicile ilustrate sau descrise ca parte a unei modalităţi de realizare pot să fie utilizate în cadrul altei modalităţi de realizare pentru a produce de asemenea o altă modalitate de realizare.

[0020] În cadrul următoarei descrieri detaliate, numeroase detalii specifice sunt expuse pentru a pune la dispoziție o completă înțelegere a subiectului tratat. Cu toate acestea, va fi de la sine înțeles pentru specialiştii din domeniu că subiectul trata poate să fie pus în practică fără aceste detalii specifice. În alte cazuri, metode, aparate sau sisteme care ar putea să fie cunoscute de către specialiştii din domeniu nu au mai fost descrise pentru a nu eclipsa subiectul tratat.

[0021] Figura 2 este o diagramă care prezintă un dispozitiv de calcul ilustrativ 202 care este utilizat pentru a transforma secvența de animație 102 în pachetul de cod 108. La dispozitivul de calcul 202 se poate face referire în mod alternativ ca la un sistem pentru prelucrarea datelor, dispozitiv computerizat, sau simplu „calculator". Dispozitivul de calcul 202 reprezintă un desktop, un laptop, o tabletă sau oricare alt sistem de calcul. Alte exemple de sisteme de calcul 202 includ, dar nu se limitează la acestea, servere, dispozitive mobile (PDA-uri, smartphone-uri, media playere, sisteme pentru jocuri etc.), televizoare, sisteme pentru jocuri şi set-top boxe cât şi sisteme încorporate (de exemplu, în vehicule, în aparate electrocasnice sau în alte dispozitive).

[0022] În general vorbind, dispozitivul de calcul 202 întruchipează unul sau mai multe elemente de hardware procesatoare de date care implementează un motor pentru codarea animației 104. Motorul pentru codarea animației 104 face ca dispozitivul de calcul 202 să analizeze datele de timeline 106 care definesc secvența de animație 102 şi generează pachetul de cod 108 care reprezintă secvența de animație 102. în cadrul acestui exemplu, secvența de animație 102 este mişcarea mingii 103 de-a lungul acestei etape, dar în practică secvențele de animație pot să fie mult mai complexe. După cum a fost remarcat mai sus, pachetul de cod este generat astfel încât, atunci când codul este prelucrat de către aplicație de rendering, capacitățile de animație native şi de scripting ale aplicației de rendering sunt invocate pentru a furniza secvența de animație prin extragerea cadrelor cheie, tranziționarea între cadrele cheie şi utilizarea primitivelor de animație aplicabile pentru asset-urile vizuale din cadrele cheie.

[0023] Motorul pentru codarea animației 104 poate să fie pus în aplicare în hardware-ul accesibil prin intermediul, sau ca o parte a, elementului pentru

prelucrarea datelor (de exemplu, ca un circuit integrat specific pentru aplicație, (ASIC), dispozitiv de logică programabilă (de exemplu, PLA-uri, FPGA-uri etc.)). Ca un alt exemplu, motorul pentru codarea animației 104 poate să fie pus în aplicare folosind software sau firmware care configurează funcționarea unui procesor sau a procesoarelor.

[0024] În cazul exemplului prezentat în Figura 2, dispozitivul de calcul 202 întruchipează un element de hardware pentru prelucrarea datelor care are în componență unul sau mai multe procesoare 204 şi un mediu citibil pe calculator (memoria 206) interconectate prin intermediul unui interconector 208, care reprezintă magistrale interne, conexiuni şi altele asemănătoare. Interconectorul 208 conectează de asemenea la componentele de I/E 210, cum ar fi magistrala serială universală (USB), VGA, HDMI, conexiuni I/E seriale şi de alt tip către alt hardware al sistemului de calcul. Hardware-ul are de asemenea în componență unul sau mai multe afişaje 212. Este de la sine înțeles că dispozitivul de calcul poate să includă alte componente, cum ar fi dispozitive de stocare, dispozitive de comunicație (de exemplu, Ethernet, componente radio) şi alte componente de I/E cum sunt difuzoarele, un microfon şi altele asemănătoare. În general vorbind, interconectorul 208 este utilizat pentru a stoca pachetul de cod generat, cum ar fi în cadrul unui hard drive local, stocare de rețea şi/sau pentru a trimite mai departe codul generat prin intermediul unei conexiuni de rețea către o destinație. Intrarea este pusă la dispoziție prin intermediul dispozitivelor de intrare cortespunzătoare cum ar fi un mouse, o tastatură, o interfață de touch-screen etc.

[0025] Mediul citibil pe calculator 206 poate să aibă în componență RAM, ROM sau o altă memorie şi în cadrul acestui exemplu încorporează un mediu de dezvoltare 214 şi motorul pentru codarea animației 104. Mai general, mediul de dezvoltare 214 este prevăzut ca un exemplu al unei sau al mai multor aplicații sau procese care utilizează sau includ motorul pentru codarea animației 104. De exemplu, mediul de runtime 214 poate să aibă în componență o aplicație de dezvoltare cum ar fi Adobe® Flash® Professional, disponibilă de la Adobe Systems Incorporated şi modificată corespunzător pentru a include sau pentru a utiliza motorul 104 pentru codarea animației.

[0026] După cum este arătat aici, mediul de dezvoltare 214 pune la dispoziție o interfață de utilizator 216 care include un stage 218 şi un timeline 220. Stage-ul 218 poate să fie folosit pentru a aranja unul sau mai multe obiecte care trebuie să fie animate cu timeline-ul 220 furnizând o reprezentare vizuală a datelor de timeline 106 şi utilizabilă pentru a selecta intervale de timp. De exemplu, timeline-ul 220 poate să fie folosit pentru a selecta diferite cadre sau alte unități de indice de timp, cu obiecte animate şi alte elemente (de exemplu, elemente de scripting) poziționate în poziții diferite în cadre diferite. În unele dintre punerile în aplicare, mediul de dezvoltare 214 poate să suporte tweening şi alte operații astfel încât un utilizator nu are nevoie să specifice fiecare poziție intermediară a unui obiect – în loc de aceasta, utilizatorul poate să specifice pozițiile de început şi de sfârşit în respectivele cadre cheie şi/sau o cale dorită, mediul de dezvoltare 214 manevrând detaliile în legătură cu corecta poziționare a obiectului(elor) între cadre fie în timpul design-ului fie prin includerea datelor în fişierul de ieşire rezultant pentru a invoca tranzitii în timpul execuției prin intermediul unei aplicații de runtime.

[0027] De exemplu, un utilizator poate să specifice o poziție de început şi de sfârşit pentru mingea 103 şi mediul de dezvoltare 214 generează codul de bait executabil/interpretabil care invocă funcționalitatea unui player runtime pentru deplasarea lină a mingii către solul G şi de la solul G, asigură săritura prezentată la 103B şi implementează schimbarea ulterioară de culoare şi mişcare la 103C (care nu este vizibilă în cadrul Figurii 2). Cu toate că acest exemplu prezintă un stage şi un timeline, un mediu de dezvoltare poate să suporte alte metode pentru a primi intrarea care specifică o secvență de animație. De exemplu, poate să fie prevăzută vederea unui cod de sursă, cu animația specificată prin intermediul unei sintaxe corespunzătoare pentru a specifica poziția şi mişcărea obiectelor.

[0028] În cazul acestui exemplu, motorul pentru codarea animației 104 este arătat ca o parte a unui mediu de dezvoltare pentru client. Totuşi, motorul pentru codarea animațieie 104 poate să fie pus în aplicare şi la un server de la distanță. De exemplu, un serviciu de web poate să găzduiască motorul 104 pentru codarea animației şi să asigure un front end pentru client pentru a furniza o interfață de utilizator prin intermediul căreia un utilizator poate defini secvența de animație. Ca un alt exemplu, motorul pentru codarea animației 104 poate să fie pus în aplicare ca

o parte a unui serviciu de web care primeşte datele de timeline 106 de la un client şi trimite ca răspuns pachetul de cod 108.

[0029] Ca un exemplu suplimentar, motorul pentru codarea animației 104 poate să acceseze datele de timeline 106 din stocare ca răspuns la o cerere a clientului pentru animație. Pe baza informației din solicitarea venită de la client (de exemplu, un agent solicitant de utilizator de browser), motorul pentru codarea animației 104 poate să optimizeze pachetul de cod pentru o platformă de client specifică. De exemplu, în cazul în care un anumit browser sau o altă aplicație de rendering nu suportă SVG-ul, motorul pentru codarea animației 104 poate să utilizeze elemente de <canvas> pentru a direcționa aplicația de rendering să extragă în loc elemente grafice. Ca un alt exemplu, în cazul în care browserul sau o altă aplicație de rendering nu suportă CSS sau animațiile (sau nici un fel de primitive de animație), elemente de scripting corespunzătoare pot să fie incluse în locul acestora. De exemplu, dacă un browser nu suportă nici un fel de animație accelerată, pot fi utilizate elemente de scripting pentru a conduce întreaga animație.

[0030] În orice caz, motorul pentru codarea animației 104 este prevăzut cu acces la datele de timeline 106 care specifică detalile secvenței de animație 102 şi foloseşte acele date pentru a genera codul care poate să fie prelucrat prin intermediul unei aplicații de rendering pentru a replica secvența de animație. Figura 3 este o diagramă de flux de date 300 care prezintă în general procesul de generare a codului executat de către motorul pentru codarea animației 104 pe un sistem de calculator 202, în timp ce Figura 4 este o schemă logică de program care discută o metodă ilustrativă 400 pentru generarea codului. Figura 5 ilustrează în detaliu un exemplu de generare a codului.

[0031] După cum este arătat la 302 din Figura 3, motorul pentru codarea animației 104 începe de la datele de timeline 106 pentru a defini secvența de animație a unuia sau a mai multor obiecte animate. De exemplu, datele de timeline 106 pot să aibă în componență codul de sursă şi/sau un model de obiect al unei aplicații de runtime, cum ar fi codul de sursă al aplicației Flash®. Totuşi, este de la sine înțeles faptul că datele de timeline 106 pot să aibă în componență o altă reprezentare a unei secvențe de animație care furnizează informație despre compoziția obiectului (adică,

identitatea obiectului şi aspectul vizual) şi informaţii despre mişcare şi despre tranziţie (adică, unde şi când se află obiectul, schimbările formei obiectului etc.)

[0032] În unele dintre punerile în aplicare, datele de timeline 106 pot de asemenea să includă unul sau mai multe elemente de scripting care definesc o funcţie asociată cu cel puţin un obiect din secvenţa de animaţie. De exemplu, funcţia poate să fie aceea de a asigura un efect cum ar fi o schimbare a aspectului unui obiect animat sau un alt element de comportare (de exemplu, deschiderea unei casete de dialog) sau de a sigura un oarecare alt comportament atunci când se face click pe un obiect.

[0033] După cum este arătat la 340, aceste date sunt analizate pentru a determina asset-urile vizuale care au în componenţă obiectul(ele) animat împreună cu datele care identifică mişcarea şi/sau alte proprietăţi care variază funcţie de timp ale resurselor vizuale pe măsură ce are loc secvenţa de animaţie. Un obiect după cum este cel reprezentat în datele de timeline 106 poate, practic vorbind, să fie animat folosind un asset vizual sau mai multe asset-uri vizuale utilizate în corelaţie unele cu altele.

[0034] De exemplu, după cum a fost remarcat mai sus, mingea 103 din Figura 1 poate să fie definită folosind grafica vectorială sau grafica raster care specifică un aspect dorit al mingii. Un obiect mult mai complex poate să includă mai multe componente a căror mişcare este coordonată în animaţie – de exemplu o minge de fotbal poate să includă elemente vizuale pentru diferitele ei părţi. În plus, datele de timeline 106 pot să definească inter-relaţiile dintre asset-uri, cum ar fi încheieturile dintre braţele şi picioarele unui corp animat. De exemplu, filmul definit de către datele de timeline poate să includă timeline-uri nested – adică, componente dintr-un timeline care sunt ele însele reprezentate ca o mulţime de alte componente. De exemplu, roţile unei biciclete pot să aibă propriul timeline care defineşte comportamentul rotaţiei, roţile fiind incluse într-un timeline principal care defineşte mişcarea întregii biciclete (de exemplu, corp şi roţi).

[0035] Prin analizarea a modului în care se mişcă asset-urile vizuale de bază sau, altfel spus, variază funcţie de timp, motorul pentru codarea animaţiei 104 poate să determine o primă porţiune de secvenţă de animaţie care poate să fie reprezentată

folosind un set de asset-uri vizuale animate folosind primitive de animaţie şi o a doua porţiune a secvenţei de animaţie poate să fie reprezentată prin utilizarea unui limbaj de scripting pentru a controla tranziţiile dintre cadrele care au fost renderate în întregime. Motorul 104 produce o reprezentare a animaţiei ca pe un set de asset-uri vizuale, primitive de animaţie corespondente şi cadre renderate după cum este prezentat la 306.

[0036] După cum este prezentat la 308, pachetul de cod 108 poate să fie generat prin selectarea statement-urilor de cod pentru aplicaţia de rendering care trebuie să proceseze pachetul de cod 108. De exemplu, în unele puneri în aplicare pachetul de cod este prevăzut ca un fişier HTML împreună cu o foaie de stil conformă – CSS3 şi un element de date structurate JSON. Fişierul HTML poate să definească aspectul cadrelor renderate şi poate de asemenea să face referire la asset-urile vizuale împreună cu statement-urile pentru a invoca stilurile definite în foaia de stil. Foaia de stil poate să fie populată cu definiţiile de stil pentru primitivele de animaţie corespondente. Elementul JSON poate să includă identificarea cadrelor cheie şi parametrilor de timing astfel încât un script aplicabil poate să fie utilizat pentru a afişa secvenţa de animaţie. Scriptul care defineşte afişajul şi operaţiile de timing executate în cadrul scriptingului poate să fie inclus în sau la el se poate face referinţă de către un fişier HTML.

[0037] Atunci când o aplicaţie de rendering, cum ar fi un browser care procesează pachetul de cod 108, aplicaţia de rendering poate efectueze scriptul pentru a rendera cadrele în ordine, utilizând capacităţile de animaţie nativă care se bazează pe definiţiile de stil pentru porţiunea(ile) secvenţei de animaţie care corespunde cu primitivele (în caz că există) suportate de către aplicaţia de rendering.

[0038] În plus, după cum este arătat la 308, atunci când pachetul de cod 108 este generat, motorul pentru codarea animaţiei poate de asmenea să includă un element de scripting corespondent (sau elemente) pentru a pune în aplicare funcţia de scripting din datele de timeline 106. De exemplu, în cazul în care un element de scripting inclus în datele de timeline 106 a pus în aplicare o funcţie pentru a furniza o fereastră de pop-up sau un alt efect atunci când se face click pe un obiect ţintă din secvenţa de animaţie 102, motorul pentru codarea animaţiei 104 poate să includă

elementul(ele) de scripting corespondent în pachetul de cod 108 şi asociat cu elementul(ele) de markup care reprezintă obiectul ţintă în pachetul de cod 108.

[0039] Figura 4 este o schemă logică de program care prezintă o metodă ilustrativă 400 perntru generarea unui pachet de cod 108 care poate să fie efectuată prin intermediul motorului pentru codarea animaţiei 104 din figurile 1-2 în conformitate cu fluxul de date din Figura 3.

[0040] Blocul 402 reprezintă accesarea datelor care definesc o secvenţă de animaţie, secvenţa de animaţie înfăţişând în timp mişcarea a cel puţin unui obiect. De exemplu, secvenţa de animaţie poate să fie definită în termenii unei poziţii pe un stage pentru un obiect sau pentru fiecare dintr-o mulţime de obiecte şi valorile de indice de timp. Ca un exemplu particular, un mediu de dezvoltare poate să menţină un model de obiect central şi de bază de cod pentru aplicaţia care se găseşte în dezvoltare. Modelul de obiect / baza de cod poate include date care reprezintă diversele componente de aplicaţie, inclusiv obiectul(ele) care trebuie să fie animate (de exemplu, mingea 103 din Figura 1) împreună cu componentele de scripting şi alte date care definesc activitatea dorită variabilă în timp a obiectului(elor) care trebuie să fie animate, inclusiv mişcarea, efectele de tranziţie şi altele asemănătoare (de exemplu, poziţii de început şi de sfârşit ale mingii 103, un număr de cadre în timpul cărora are loc deplasarea, parametri care definesc modul în care se deformează mingea în porţiunea de săritură 103B, tranziţia de culoare la 103C etc.).

[0041] Blocul 404 reprezintă date de accesare care identifică primitive de animaţie suportate de către un limbaj de markup. De exemplu, motorul pentru codarea animaţiei 104 poate să fie hard-coded pentru a recunoaşte câteva dintre sau un întreg set de operaţii de animaţie care pot să fie invocate prin utilizarea unei foi de stil, cum ar fi o foaie de stil CSS3. Ca un alt exemplu, motorul pentru codarea animaţiei 104 poate să acceseze în mod selectiv diferite seturi de primitive suportate de diferite limbaje de foaie de stil, diferite aplicaţii de rendering şi de altele asemenea pe baza unui format de ieşire dorit. În cazul în care pachetul de cod trebuie să fie personalizat pe baza capacităţilor browserului sau ale aplicaţiei de rendering, atunci blocul 404 reprezintă determinarea primitivelor care sunt suportate de către un anumit browser sau aplicaţie de rendering.

[0042] Blocul 406 reprezintă analizarea datelor care definesc secvența de animație pentru a determina un set de cadre şi de asset-uri vizuale cu primitivele de animație corespondente utilizabile pentru a reprezenta mişcarea în timp a cel puțin unui obiect. De exemplu, motorul pentru codarea animației 104 poate să utilizeze datele care descriu pozițiile obiectului în timp şi/sau definirea datelor activității variabile funcție de timp a obiectelor pentru a identifica o primă porțiune a secvenței de animație în care mişcarea/comportamentul obiectului poate să se potrivească cu primitivele de animație. Obiectul(ele) animate pot să fie desfăcute în una sau mai multe asset-uri vizuale, fiecare asset vizual fiind animat individual într-un astfel de mod încât timing-ul original al porțiunii secvenței de animație este păstrat împreună cu poziția relativă intenționată a obiectului(elor) animate.

[0043] Motorul pentru codarea animației 104 poate de asemenea să utilizeze date pentru a determina o a doua porțiune a secvenței de animație care poate să fie reprezentată prin intermediul utilizării unui limbaj de scripting pentru a controla tranzițiile dintre cadrele cheie complet renderate. De exemplu, motorul pentru codarea animației 104 poate să renunțe la utilizarea cadrelor renderate în cazul în care setul de primitive de animație nu suportă mişcarea/comportamentul obiectului din secvența de animație. Ca un exemplu particular, porțiunea de săritură 103B a secvenței de animație 102 şi/sau schimbarea de culoare a mingii 103 pot să nu aibă primitive de animație omoloage. În felul acesta, motorul pentru codarea animației 104 poate să determine ca acele porțiuni ale secvenței de animație 102 trebuie să fie reprezentate cu utilizarea cadrelor renderate direct.

[0044] Blocul 408 reprezintă generarea unui pachet care face ca o aplicație de rendering să furnizeze secvența de animație prin utilizarea unui limbaj de scripting. Pachetul are în componență codul de markup care defineşte un aspect renderat al cadrelor cheie împreună cu referințele markup-ului pentru setul de asset-uri vizuale pentru porțiunea(ile) secvenței de animație care poate să fie reprezentată folosind primitive de animație.

[0045] O foaie de stil defineşte primitivele de animație corespondente ca stiluri de aplicat asset-urilor vizuale. Pachetul este generat astfel încât, atunci când markup-ul de cod este prelucrat de către aplicația de rendering, aplicația de rendering utilizează un motor de scripting (după cum este acesta configurat de către un obiect

de date structurate) pentru a coordona rendering-ul porţiunii(unilor) secvenţei de animaţie reprezentate utilizând primitivele de animaţie şi porţiunea de secvenţă reprezentată direct folosind cadrele renderate. De exemplu, poate să fie asigurat un set de fişiere după cum a fost discutat mai sus în legătură cu Figura 1, setul de fişiere incluzând codul de markup care poate să fie renderat de către un browser şi făcând referinţă la o foaie de stil.

[0046] Figura 5 este o schemă logică de program care prezintă paşii ilustrativi într-o metodă 500 pentru analizarea datelor care definesc o secvenţă de animaţie şi care generează un pachet de cod 108. De exemplu, fluxul prezentat în cadrul Figurii 5 poate să fie utilizat de către motorul pentru codarea animaţiei 104 ca şi cum ar fi pus în aplicare de către un dispozitiv cum ar fi un calculator 202 din Figura 2.

[0047] Blocul 502 reprezintă accesarea datelor de timeline multilayer şi layer de compositing pentru a identifica conţinutul renderat al cadrelor. De exemplu, runtime-ul Flash® amintit mai sus poate să utilizeze fişierele sursă în formatul FLA pentru a defini un film care poate să conţină mai multe scene şi simboluri, care au fiecare propriul timeline. Fiecare timeline poate avea mai multe layere. Întregul film are un frame rate utilizat pentru a calcula momentul la care este afişat următorul cadru. Conţinutul unui cadru al filmului poate să fie definit prin compositingul conţinutului fiecărui layer de jos până sus pentru a determina un aranjament al caracteristicilor vizuale (şi al altora) la fiecare punct în decursul timpului de-a lungul filmului. Conţinutul cadrului poate să fie în general definit ca având în componenţă forme – muchii şi stiluri de umplere utilizate pentru a rendera forma. Pe parcusul procesului de conversie, formele pot să fie convertite în grafice cum sunt fişierele SVG sau <canvas> HTML5 sau alte comenzi pentru extragerea markup-ului. Graficele raster pot de asemenea să fie utilizate în unele modalităţi de realizare.

[0048] Blocul 504 reprezintă selectarea şi sortarea cadrelor cheie ale filmului în ordinea apariţiei lor. De exemplu, datele de timeline 106 care definesc filmul pot să definească cel puţin unele porţiuni din secvenţa de animaţie utilizând cadre cheie care prezintă un aspect renderat al animaţiei la un anumit moment, cu porţiuni de animaţie între cadrele cheie care trebuie „umplute" de către aplicaţia de runtime în conformitate cu alte date incluse în datele de timeline 106 pentru a provoca

rendering-ul unei întregi porţiuni din secvenţa de animaţie. De exemplu, secvenţa de animaţie 102 din Figura 1 poate să fie definită de către un cadru cheie care reprezintă o poziţie iniţială a mingii 103, un cadru cheie care indică o poziţie a porţiunii de săritură 102B şi un cadru cheie care indică poziţia finală a mingii 103. Detaliile porţiunilor 103A, B şi C pot să fie definite prin intermediul altor date din datele de timeline 106 (de exemplu, date de cale care indică cum se mişcă mingea 103, date de transformare care definesc cum este deformată mingea 103 în timpul coliziunii ei cu G, date gradient despre cum este definită schimbarea de culoare etc.)

[0049] Blocul 506 reprezintă generarea unui container de markup pentru fiecare secvenţă de cadre cheie. De exmplu, un element HTML „ " poate să fie creat pentru fiecare secvenţă de două cadre cheie. Ca un exemplu, să presupunem că porţiunea 103B a secvenţei de animaţie 102 începe la cadrul 10 şi se termină la cadrul 20. Un element poate să fie creat pentru cadrele de la 1 la 9 (adică porţiunea 103A) şi un alt element pentru cadrele de la 10 la 20 (porţiunea 103B). continuând cu exemplul, un alt element poate să fie creat pentru cadrele care reprezintă porţiunea 103C.

[0050] Datele de timing pot să fie asociate cu obiectul de date structurate pentru a indica pentru limbajul de scripting care dintre elementele din codul de markup corespunde cu cadrele cheie ale mulţimii de cadre şi pentru a indica când trebuie să fie vizibile cadrele. De exemplu, obiectul de date structurate poate să includă numele de element şi date de timing pentru a controla când JavaScript sau un alt motor de scripting este utilizat pentru a schimba o proprietate de stil care controlează vizibilitatea diverselor elemente astfel încât ele sunt vizibile în secvenţă. Continuând cu exemplul de mai sus, datele pot să fie incluse într-un obiect JSON astfel încât elementul care corespunde cu cadrele 1 – 10 este afişat pentru o primă perioadă de timp şi apoi este afişat elementul care corespunde cu cadrele 10 – 20 (urmat de următoarele cadre).

[0051] Blocul 506 reprezintă de asemenea includerea unui markup pentru a defini aspectul renderat al fiecărui cadru cheie. De exemplu, motorul pentru codarea animaţiei 104 poate să includă mai multe elemente de cod de markup cum sunt elemente <canvas> cu comenzi de extragere corespunzătoare pentru a recrea un

aspect renderat al fiecărui cadru în acea porțiune a secvenței de animație. Asset-uri vizuale pot să fie poziționate în cadrele cheie dacă este potrivit. De exemplu, SVG sau alte grafice pot să fie utilizate pentru a reduce extragerea şi alți timpi de prelucrare.

[0052] Blocurile 508A-508B prezintă cum pot să fie utilizate diferite porțiuni din secvența de animație bazat pe dacă porțiunea poate să fie reprodusă folosind primitive de animație. De exemplu, motorul pentru codarea animației 104 poate să analizeze mişcarea obiectului definită în datele de timeline 106 şi/sau se poate baza pe definițiile mişcării incluse în datele 106 pentru a determina dacă mişcarea unuia sau mai multor obiecte se mapează pe un primitiv.

[0053] De exemplu, platforma Flash® suportă diferite tipuri de animație, inclusiv un Shape Tween care se bazează pe datele care definesc două forme, datele de timeline 106 incluzând o comandă pentru player ca să interpoleze cadrele care vin între acele două forme (de exemplu, prin utilizarea capacităților de morphing incluse în player). Platforma Flash® suportă de asemenea un Classic Tween care animează transformările şi proprietățile aplicate la un simbol între două momente de timp. Playerul trebuie să interpoleze transformările aplicate în cadre între două puncte cheie. Platforma Flash® suportă de asemenea un Motion Tween similar cu Classic Tween, care utilizează punte cheie şi alte date pentru a preciza mişcarea.

[0054] În cadrul unei puneri în aplicare, motorul pentru codarea animației 104 determină dacă o porțiune a animației este specificată la un „Shape Tween". În cazul în care este, motorul pentru codarea animației 104 utilizează abordarea la blocul 508A pentru acea porțiune şi converteşte acea porțiune a animației la o secvență a cadrelor renderate direct şi determină sintaxa corespunzătoare (de exemplu, comenzile elementului <canvas>) pentru a extrage fiecare cadru. De exemplu, pentru porțiunea 103B a secvenței de animație 102, fiecare cadru al deformării şi săriturii mingii 103 poate să fie renderat şi analizat pentru a determina <canvas> sau alte comenzi pentru a rendera cadrul. Elementul <canvas> pentru cadrul renderat poate să fie inclus într-un element sau într-un alt container, datele de timing şi de tranziție fiind incluse în obiectul de date structurate astfel încât cadrele renderate să fie prezentate în ordine.

[0055] Blocul 508B prezintă cum se adresează motorul pentru animație 104 porțiunilor secvenței de animație care pot să fie reprezentate utilizând un primitiv de animație care poate să fie definit ca un stil. În particular, motorul pentru codarea animației include un element din codul de markup care reprezintă un asset vizual în secvența de animație, iar motorul pentru codarea animației include un stil în foaia de stil pentru aplicarea primitivului de animație la assetul vizual.

[0056] De exemplu, pentru Classic Tween şi Motion Tween, motorul pentru codarea animației 104 selectează un CSS sau un alt primitiv de animație şi include o definiție de stil în foaia de stil. Motorul pentru codarea animației 104 include de asemenea date de timing în obiectul de date structurate pentru a declanşa şi coordona rendering-ul utilizând primitive de animație. De exemplu, diferite asseturi vizuale (de exemplu, fişiere SVG) pot să fie incluse în containerele încorporate în cadrele cheie (de exemplu, containere în elemente <canvas> într-un pentru cadrul cheie), numele containerului fiind utilizat pentru a aplica respectivele stiluri la assetuuri. Animațiile pot să fie declanşate prin includerea parametrilor în obiectul de date structurate pentru a declanşa primitivul(ele) atunci când containerul pentru cadrul cheie este făcut vizibil.

[0057] Stilurile pot să fie ele însele definite utilizând valori de parametri care păstrează timingul relativ al animațiilor diverselor asset-uri şi relațiile spațiale ale asset-urilor pe parcursul animației. De exemplu, anumite animații pot să fie întârziate în raport cu altele şi/sau repetate, valorile de durată pentru primitivele de animație fiind utilizate pentru a controla viteza relativă dintre animații. Valorile coordonate pot să fie incluse în definițiile de stil astfel încât aranjamentul asseturilor vizuale să rămână adevărat cu originalul secvenței de animație pe măsură ce asset-urile sunt translatate, rotite, deformate şi altele asemenea. Asset-urile vizuale pot fi ele însele incluse ca fişiere (de exemplu, fişiere de imagine raster, fişiere de grafică vectorială proporțională) sau pot fi definite ca elemente din codul de markup.

[0058] De exemplu, în cazul în care asset-ul translatează de-a lungul stage-ului, atunci motorul pentru codarea animației poate să selecteze un primitiv de animație care, atunci când este prelucrat de către aplicația de rendering, face ca aplicația de

rendering să mişte asset-ul vizual într-o a doua poziţie în interfaţa aplicaţiei de rendering care corespunde cu a doua poziţie a assetului vizual de pe stage.

[0059] Ca un exemplu particular, primitivul webkit-transform poate să fie selectat pentru utilizarea ca un stil aplicat unui asset vizual care reprezintă mingea 103 în porţiunile 103A şi 103C ale secvenţei de animaţie 102, valorile pentru a defini poziţiile de început şi de sfârşit şi o rată dorită pentru translaţie putând fi determinate. În cazul în care mingea 102 se şi roteşte, un primitiv de rotaţie poate să fie selectat prin intermediul unei analize similare. Tranziţiile (de exemplu, fade-in, fade-out), deformările şi alte manipulări suportate ca primitive de animaţie pot să fie de identificate ca atare.

[0060] Astfel, pentru porţiunea 103A a secvenţei de animaţie 102, un cadru cheie poate să fie renderat şi apoi mingea 103 poate să fie animată utilizând un webkit-transform sau un alt primitiv, pentru a fi urmat de rendering-uri directe ale cadrelor care reprezintă porţiunea 103B, urmate de un cadru cheie care reprezintă începutul porţiunii 103C, mişcarea mingii fiind asigurată prin utilizarea unui alt primitiv. Schimbarea de culoare a mingii 103 poate să fie prevăzută prin utilizarea unui asset vizual diferit, al unei comenzi de extragere sau prin renderingul direct al tranziţiei de culoare utilizând mai multe cadre în porţiunea 103C până când este obţinut aspectul final.

[0061] Blocul 510 reprezintă includerea elementelor de scripting corespunzătoare sau a referinţelor în elemente de scripting pentru a controla aspectul containerelor şi timingul primitivelor de animaţie. De exemplu, un fişier JavaScript poate să fie încorporat în sau la el se poate face referire de către codul de markup care include containerele , elementele <canvas> şi referinţele la asset-uri vizuale. Fişierul JavaScript poate să fie utilizat pentru a afişa în ordine cadrele cheie şi cadrele renderate, tranziţia dintre cadre şi pentru a declanşa primitivele de animaţie pentru a orchestra animaţia în conformitate cu datele dintr-un obiect JSON (sau un alt element de date structurate) inclus în pachetul de cod.

[0062] O secvenţă de animaţie poate să fie divizată în mai multe porţiuni pentru a fi reprezentată utilizând primitivele de animaţie şi cadrele renderate direct. În unele

puneri în aplicare, porțiuni animate mai scurte pot să fie compozite – de exemplu, mai multe classic tween-uri pot să fie unite împreună într-o singură animație pe baza unui primitiv. Raportul dintre porțiunile animate acționate prin primitive față de porțiunile animate renderate direct poate să varieze şi după cum a fost remarcat mai sus poate chiar să depindă de dacă pachetul de cod este generat personalizat pentru o anumită aplicație de rendering.

[0063] O punere în aplicare poate să utilizeze cu precădere sau numai cadre cheie renderate direct, dar performanțele pot să fie afectate cel puțin pe unele dintre platforme. Prin utilizarea primitivelor de animație bazate pe stil, motorul pentru codarea animației 104 poate să furnizeze codul care să impulsioneze grafica Power a aplicațiilor de rendering. De exemplu, animațiile bazate de CSS pot să fie renderate direct în conformitate cu codul nativ (de exemplu, cod binar optimizat al aplicației) sau utilizând procesele de rendering accelerat ale unității de procesare grafică (GPU). Cu toate acestea, efecte mai complexe pot să fie păstrate prin utilizarea de secvențe de cadre renderate direct.

[0064] Blocul 510 reprezintă de asemenea includerea unuia sau mai multor elemente de scripting pentru a furniza o funcție scriptată definită în datele de timeline 106. De exemplu, platforma de Flash® suportă evenimente scriptate, cum sunt funcțiile definite folosind Actionscript™ sau JavaScript. În particular, evenimentele scriptate pot să includă efecte vizuale cum ar fi schimbările de aspect ale obiectului ca răspuns la click-uri, mouseover sau alte evenimente; alte efecte cum sunt ferestrele popup, solicitările rețelei etc; sau alt comportament personalizat. Motorul pentru codarea animației 104 poate să depisteze care obiect(e) din datele de timeline 106 este asociat cu funcționalitatea scriptată în datele de timeline 106 şi să determine elementele corespondente în pachetul de cod 108 (de exemplu, asseturile vizuale corespondente, containere, elemente <canvas> etc.). Atunci când este generat pachetul de cod 108, elementele de scripting corespondente pot să fie incluse în pachetul de cod pentru a furniza funcția scriptată.

[0065] De exemplu, în cazul în care datele de timeline 106 includ o funcție JavaScript asociată cu un obiect grafic, atunci funcția JavaScript poate să fie inclusă în sau la ea se poate face referire prin codul de markup în pachetul de cod 108,

obiectul şi referinţele variabile fiind actualizate pentru a reflecta procesul de conversie astfel încât Java SAcript inclusă să facă referire la elementul(ele) utilizat pentru a reprezenta obiectul grafic.

[0066] Figura 6 este o schemă bloc care prezintă o arhitectură ilustrativă 600 pentru un mediu de dezvoltare 214 care utilizează un motor pentru codarea animaţiei 104 configurat în conformitate cu prezentul subiect. În cadrul acestui exemplu, mediul de dezvoltare a aplicaţiei 214 include de asemenea un modul UI 602 utilizat pentru a furniza o interfaţă grafică de utuilizator şi pentru a primi o intrare de utilizator. De exemplu, modulul UI poate să rendereze interfaţa grafică de utilizator 216 din Figura 2, inclusiv stage-ul de design 218, vederea de timeline 220 şi alte interfeţe (de exemplu, interfaţa de vedere de cod etc.) şi să populeze interfaţa cu secvenţa de animaţie 102 pe măsură ce secvenţa este definită/editată.

[0067] Modulul manager de modul de obiect 604 poate să acceseze datele stocate care reprezintă un model de obiect central şi baza de cod 606 pentru aplicaţia aflată în dezvoltare. Modelul de obiect/baza de cod poate să includă date care reprezintă diversele componente de aplicaţie, inclusiv elemente de medii, componente de scripting şi altele asemenea şi este reprezentativ pentru datele de timeline 106 utilizate de către modulul pentru codarea animaţiei 104. De exemplu, modulul 604 poate să stocheze grafică vector, grafică raster sau alte grafici care reprezintă mingea 103 împreună cu datele care definesc poziţia mingii 103 în diverse cadre, împreună cu efectele de mişcare dorite pe măsură ce mingea 103 schimbă poziţia în diferitele cadre.

[0068] După cum a fost discutat mai sus, motorul pentru codarea animaţiei 104 poate să utilizeze date de timeline 106 pentru a descompune o secvenţă de animaţie într-o mulţime de asseturi vizuale şi pentru a selecta primitivele de animaţie corespondente pentru a fi aplicate acelor asset-uri vizuale în scopul de a replica secvenţa de animaţie prin intermediul pachetului de cod 108 procesat de către o aplicaţie de rendering, cadrele cheie renderate direct utilizate pentru una sau pentru mai multe porţiuni ale animaţiei nefiind reproduse prin utilizarea de primitive. După cum este cerut, componentele de scripting aplicate elementelor de timeline pot să fie convertite şi incluse în pachetul de cod 108.

[0069] Motorul pentru codarea 104 poate, de exemplu, să desfăşoare analiza în conformitate cu Figurile de la 3 la 5 discutate mai sus. Aplicaţia de dezvoltare 212 include de asemenea modulul de compilare 608. Modulul compilator 608 poate să utilizeze modelul de obiect / baza de cod pentru a produce codul executabil sau interpretabil al aplicaţiei aflată sub dezvoltare. Codul de ieşire poate, de exemplu, să aibă în componenţă fişiere SWF sau AIR pentru execuţia Adobe® Flash® sau AIR®, fişiere pentru execuţia în alt mediu runtime, fişiere pentru execuţia într-un sistem de operare, sau altele asemenea.

[0070] Este de la sine înţeles că prezentul subiect poate să fie folosit indiferent de formatul sau tipul aplicaţiei aflată sub dezvoltare, iar construcţia şi utilizarea compilatoarelor, linkurilor şi componentelor de packaging corespunzătoare (de exemplu, pentru compatibilitatea cross-platform) va fi la îndemâna specialistului din domeniu. Aceasta poate să dea unui dezvoltattor posibilitatea de a genera o secvenţă de animaţie o dată şi apoi să scoată secvenţa în mai multe formate diferite (de exemplu, în HTML/CSS3 şi ca o aplicaţie Flash®).

[0071] Motorul pentru codarea animaţiei 104 este prezentat, în cadrul acestui exemplu, ca integrat în mediul de dezvoltare 214. Este de la sine înţeles că motorul pentru codarea animaţiei 104 poate să opereze independdent de mediul de dezvoltare 214. De exemplu, motorul pentru codarea animaţiei 104 poate să fie implementat cu propriul modul UI utilizat pentru a selecta fişiere care conţin date de timeline 106 şi convertesc aceste fişiere într-un pachet de cod 108. În plus, motorul pentru codarea animaţiei 104 poate să fie prevăzut ca un serviciu de web sau ca un alt serviciu găzduit sau poate să fie folosit la un dispozitiv (de exemplu, un dispozitiv de client) pentru a converti fişierele care sosesc într-un format utilizabil de către dispozitiv.

Consideraţii generale

[0072] Unele porţiuni ale descrierii detaliate au fost prezentate în termeni de reprezentare simbolică sau algoritmică a operaţiunilor pe biţi de date sau semnale digitale stocate în interiorul memoriei unui sistem de calcul, cum este o memorie de calculator. Aceste descrieri sau reprezentări algoritmice sunt exemple de tehnică

folosită de către specialiştii din domeniul prelucrării datelor pentru a transmite substanța muncii lor altor specialişti din domeniu. Un algoritm este considerat aici şi în general a fi o secvență de operațiuni auto-suficientă sau procesare similară cu aceasta care conduce la un rezultat dorit. În acest context, operațiunile şi procesarea implică manipularea fizică şi mărimi fizice.

[0073] În mod obişnuit, deşi nu este necesar, astfel de mărimi pot lua forma unor semnale electrice sau magnetice care au posibilitatea de a fi stocate, transferate, combinate, comparate sau în alt fel folosite. S-a dovedit convenabil uneori, în principal din motive de utilizare comună, să se facă referire la astfel de semnale ca la biți, date, valori, elemente, simboluri, litere, termeni, numere, numerale sau altele asemănătoare. Trebuie totuşi să se înțeleagă că toate acestea şi termenii similari trebuie să fie asociați cu mărimi fizice corespunzătoare şi sunt pur şi simplu etichete convenabile.

[0074] În afară de cazul în care este afirmat altceva, după cum reiese din discuția anterioară, se consideră că în întreaga această specificație discuțiile care utilizează termeni precum „prelucrare", „calculare", „care calculează", „care determină" sau altele asemenea se referă la acțiuni sau la procese ale unei platforme de calcul, cum ar fi unul sau mai multe calculatoare şi/sau un dispozitiv sau dispozitive de calcul electronice similare, care folosesc sau transformă date reprezentate ca mărimi fizice electronice sau magnetice din memorii, registrii sau alte dispozitive de stocare a informației, dispozitive de transmisie, sau dispozitive de afişare ale platformei de calcul.

[0075] Deşi mai multe exemple au prezentat dispozitive mobile, diversele sisteme care au fost discutate aici nu sunt limitate la nicio arhitectură sau configurație particulară de hardware. Un dispozitiv de calcul poate să includă orice aranjament corespunzător de componente care furnizează un rezultat condiționat de una sau de mai multe intrări. Dispozitive de calcul potrivite includ sisteme multifuncționale de calculator bazate pe microprocesoar care accesează software-ul stocat, care programează sau configurează sistemul de calcul dintr-un aparat de calcul de destinație generală într-un aparat de calcul specializat prin punerea în aplicare a uneia sau mai multor modalități de realizare a prezentului subiect. Oricare limbaj

corespunzător de programare, de scripting sau un alt tip de limbaj sau combinație de limbaje poate să fie folosit pentru a pune în aplicare ştiința conținută aici în software-ul care urmează să fie folosit pentru programarea sau configurarea unui dispozitiv de calcul.

[0076] Un dispozitiv de calcul poate să acceseze unul sau mai multe medii ne-tranzitorii citibile pe calculator care încorporează instrucțiuni citibile pe calculator care, atunci când sunt executate de către cel puțin un calculator, fac ca acel cel puțin un calculator să implementeze una sau mai multe din modalitățile de realizare ale prezentului subiect. Atunci când este utilizat software-ul, software-ul poate să aibă în includă una sau mai multe componente, procese şi/sau aplicații. În plus sau alternativ la software, dispozitivul(ele) de calcul poate să aibă în componență circuite care fac dispozitivul(ele) funcțional pentru a implementa una sau mai multe metode ale prezentului subiect.

[0077] Exemplele de dispozitive de calcul includ, fără a fi limitate la acestea, servere, computere personale, dispozitive mobile (de exemplu, tablete, smartphone-uri, asistente digitale personale (PDA-uri) etc.), televizoare, set-top boxe pentru televizoare, playere portabile pentru muzică şi dispzitive electronice pentru amatori cum ar fi aparatele de fotografiat, aparatele de filmat şi dispozitivele mobile. Dispozitivele de calculat pot să fie integrate în alte dispozitive, de exemplu electrocasnice „inteligente", automobile, chioşcuri şi altele asemenea.

[0078] Modalități de realizare a metodelor descrise aici pot să fie realizate în operațiile dispozitivelor de calcul. Ordinea blocurilor prezentate în exemplele de mai sus poate să fie diferită – de exemplu, blocurile pot să fie re-ordonate, combinate şi/sau descompuse în sub-blocuri.
Anumite blocuri sau procese se pot desfăşura în paralel.

[0079] Oricare mediu sau medii citibile pe calculator ne-tranzitorii corepunzătoare pot să fie utilizate pentru a implementa sau aplica în practică subiectul descris aici, inclusiv, dar fără a se limita la, dischete, drive-uri, medii de stocare pe bază magnetică, medii de stocare optice (de exemplu, CD-ROM-uri, DVD-ROM-uri şi variante ale acestora), dispozitive de memorie flash, RAM, ROM şi altele.

[0080] Utilizarea aici a termenilor de „adaptat la" sau „configurat pentru" este înțeleasă ca limbaj deschis şi incluziv care nu exclude dispozitivele adaptate pentru sau configurate pentru a executa sarcini şi paşi suplimentari. În plus, utilizarea lui „pe baza" este înțeleasă ca fiind deschisă şi incluzivă, prin aceea că un proces, o etapă, un calcul sau o altă acțiune care "se bazează pe" una sau mai multe din condițiile sau valorile relatate poate, în practică, să fie bazată pe condiții sau valori adiționale celor care au fost enunțate. Titlurile, listele şi numerotarea incluse aici sunt numai pentru uşurarea explicației şi nu sunt intenționate ca limitative.

[0081] Chiar dacă prezentul subiect a fost descris în detaliu cu referire la anumite modalități de realizare a acestuia, este de la sine înțeles pentru specialiştii din domeniu, după parcurgerea şi înțelegerea celor anterioare că pot cu uşurință să aducă modificări la, variații la şi echivalente la aceste modalități de realizare. În consecință, este de la sine înțeles că prezenta descriere a fost prezentată numai pentru scopuri de exemplificare şi nu de limitare, şi că nu împiedică includerea unor astfel de modificări, variații şi/sau suplimentări la prezentul subiect după cum este evident pentru specialiştii din domeniu.

Revendicări:

1. O metodă pusă în aplicare pe calculator, care are în componență:
accesarea datelor care definesc o secvență de animație, secvența de animație înfățișând mişcarea în timp a cel puțin unui obiect;
accesarea primitivelor de animație care identifică datele suportate de către un limbaj de markup;
analizarea datelor care definesc secvența de animație pentru a determina o primă porțiune a secvenței de animație care poate să fie reprezentată prin intermediul utilizării unui set de asset-uri vizuale animate prin utilizarea primitivelor de animație şi o a doua porțiune a secvenței de animație care poate să fie reprezentaă prin utilizarea unui limbaj de scripting pentru a controla tranzițiile dintre cadrele care reprezintă a doua porțiune, şi
generarea unui pachet, pachetul având în componență codul de markup care face referire la asset-urile vizuale şi care defineşte un aspect al cadrelor, o foaie de stil care face referință la primitivele de animație şi un obiect de date structurate,
codul de markup generat pentru a provoca o aplicație de rendering să furnizeze secvența de animație prin utilizarea limbajului de scripting pentru a coordona rendering-ul fiecăreia din prima porțiune si a doua porțiune în conformitate cu un parametru inclus în obiectul de date structurate, prima porțiune renderată prin aplicarea primitivelor de animație ca stiluri la setul de asset-uri vizuale şi a doua porțiune renderată prin extragerea mulțimii de cadre şi tranziționarea între cadre folosind limbajul de scripting.

2. Metodă implementată pe calculator în conformitate cu rervendicarea 1, în care datele care definesc secvența de animație au în componență un element de scripting care defineşte o funcție asociată cu cel puțin un obiect şi care generează pachetul au în componență inclusiv un element de scripting corespondent pentru a face ca aplicațiea de rendering să furnizeze funcția.

3. Metodă în conformitate cu revendicarea 1, în care analizarea datelor care definesc secvența de animație are în componență determinarea faptului dacă mişcarea în timp a obiectului corespunde cu un primitiv de animație suportat de către un browser, în care, dacă mişcarea nu corespunde cu un primitiv de animație

suportat de către un browser, mişcarea este reprezentată prin utilizarea mulțimii de cadre.

4. Metodă în conformitate cu revendicarea 1, în care fiecare cadru din mulțimea de cadre este reprezentat în codul de markup folosind un element <canvas> HTML care defineşte un aspect renderat al cadrului.

5. Metodă în conformitate cu revendicarea 1, în care obiectul de date structurate indică pentru limbajul de scripting care elemente din codul de markup corespunde cu cadrele din mulțimea de cadre şi indică în plus pentru limbajul de scripting momentul în care cadrele trebuie să fie făcute vizibile.

6. Metodă în conformitate cu revendicarea 1, având în plus în componență:
utilizarea datelor care definesc secvența de animație pentru a genera o aplicație de runtime care furnizează secvența de animație.

7. Un dispozitiv computerizat, care are în componență:
un interconector de hardware; şi
un element de hardware pentru prelucrarea datelor interfațat cu interconectorul de hardware,
în care elementul de hardware pentru prelucraea datelor pune în aplicare un motor pentru codarea animației pentru a analiza datele de timeline care definesc o secvență de animație şi generează un pachet de cod,
pachetul de cod reprezentând secvența de animație care utilizează codul de markup care defineşte un aspect renderat al unei mulțimi de cadre, pachetul de cod având în componență codul de markup şi un obiect de date structurate care definesc un parametru utilizat de către un limbaj de scripting în tranziționarea dintre cadrele mulțimii de cadre.

8. Dispozitiv computerizat în conformitate cu revendicarea 7, în care codul de markup are în componență o referință la un asset vizual inclus într-un cadru iar pachetul de cod are în plus în componență o foaie de stil în casadă care defineşte un primitiv de animație ca pe un stil de aplicat asset-ului.

9. Dispozitiv computerizat în conformitate cu revendicarea 8, în care:
codul de markup are în componență o referință la un prim element container de markup care are în componență un cod care defineşte un aspect renderat al unui prim cadru,
markup-ul are în componență o referință la un al doilea element container de markup care are în componență codul care defineşte un aspect renderat al celui de-al doilea cadru şi
parametrul din obiectul de date structurate indică respectivii timpi pe durata cărora trebuie să fie vizibil conținutul primului şi al celui de-al doilea element container de markup.

10. Dispozitiv computerizat în conformitate cu revendicarea 9, în care parametrul din obiectul de date structurate controlează când trebuie să fie folosit limbajul de scrpting pentru a schimba proprietatea unui stil care controlează vizibilitatea primului şi celui de-al doilea element de container de markup.

11. Dispozitiv computerizat în conformitate cu revendicarea 9, în care codul de markup are în plus în componență o referință la un asset vizual, referința fiind inclusă într-unul dintre primul şi cel de-al doilea cadru şi asociată cu un stil care defineşte un primitiv de animație care trebuie să fie aplicat la assetul vizual.

12. Dispozitiv computerizat în conformitate cu revendicarea 8, în care motorul pentru codarea animației analizează datele de timeline care definesc secvența de animație pentru a determina dacă o porțiune din secvența de animație poate să fie reprezentată folosind un primitiv de animație care poate să fie definit ca un stil; şi
în cazul în care secvența de animație nu poate să fie reprezentată prin utilizarea unui primitiv de animație care poate să fie definibil ca un stil:
motorul pentru codarea animației include mai multe elemente de cod de markup în codul de markup, fiecare element de cod de markup fiind corespondent cu un aspect renderat al unui cadru al porțiunii de secvență de animație, iar motorul pentru codarea animației include un parametru în obiectul de date structurate pentru a controla rendering-ul mulțimii de elemente de cod de markup astfel încât să afişeze cadrele din secvență;

în care secvența de animație poate să fie reprezentată utilizând un primitiv de animație definibil ca un stil:
motorul pentru codarea animației include un element din codul de markup care reprezintă un asset vizual din secvența de animație, iar motorul pentru codarea animației include un stil din foaia de stiluri pentru a aplica primitivul de animație la assetul vizual.

13. Dispozitiv computerizat în conformitate cu revendicarea 8, în care limbajul de scripting are în componență Java Script iar obiectul de date structurate are în componență un obiect JSON (JavaScript Object Notation).

14. Dispozitiv computerizat în conformitate cu revendicarea 8, în care motorul pentru codarea animației identifică în plus un element de scripting aplicat unui element din datele de timeline şi include, în pachetul de cod, un element de scripting corespondent care să fie aplicat unui asset corespondent din codul de markup.

15. Dispozitiv computerizat în conformitate cu revendicarea 8, în care hardware-ul pentru prelucrarea datelor are în componență un procesor iar motorul pentru codarea animației are în componență o logică de program accesabilă de către procesor.

16. Dispozitiv computerizat în conformitate cu revendicarea 8, în care motorul pentru codarea animației este cuprins într-un mediu de dezvoltare configurat pentru a utiliza datele de timeline pentru a genera o aplicație de runtime care pune la dispoziție secvența de animație.

17. Produs program de calculator care are în componență un mediu citibil pe calculator ne-tranzitoriu care încorporează un cod de program, codul de program având în componență:
codul pentru furnizarea unui canvas de design şi pentru primirea intrării care specifică o secvență de animație care reprezintă un aspect variabil în timp al cel puțin unui obiect animat pe un stage;
codul pentru stocarea datelor care definesc secvența de animație;

codul pentru analizarea datelor care definesc secvența de animație şi pentru generarea unui pachet, pachetul având în componență codul de markup şi un obiect de date structurate, codul de markup definind mai multe cadre dintre care fiecare reprezintă aspectul obiectului animat pe stage la respectivul punct din timp, obiectul de date structurate incluzând un parametru care să fie folosit de către o aplicație de rendering în controlarea momentului la care fiecare cadru din mulțimea de cadre este vizibil.

18. Produs program de calculator în conformitate cu revendicarea 17, în care pachetul are de asemenea în componență o foaie de stil care invocă un primitiv de animație suportat de către aplicația de rendering, obiectul de date structurate indicând momentul în care primitivul de animație trebuie să fie aplicat unui asset vizual în unul dintre cadrele care reprezintă o porțiune din secvența de animație fără a face tranziția la un cadru diferit.

19. Produs program de calculator în conformitate cu revendicarea 18, în care assetul vizual este inclus într-o porțiune de cod de markup care defineşte un aspect renderat al unuia dintre cadre.

20. Produs program de calculator în conformitate cu revendicarea 16, având în plus în componență:
cod de program pentru generarea unei aplicații de runtime bazate pe datele care definesc secvența de animație, aplicația de runtime fiind configurată pentru a furniza secvența de animație atunci când este executată.

a-2011-00322--
07 -04- 2011

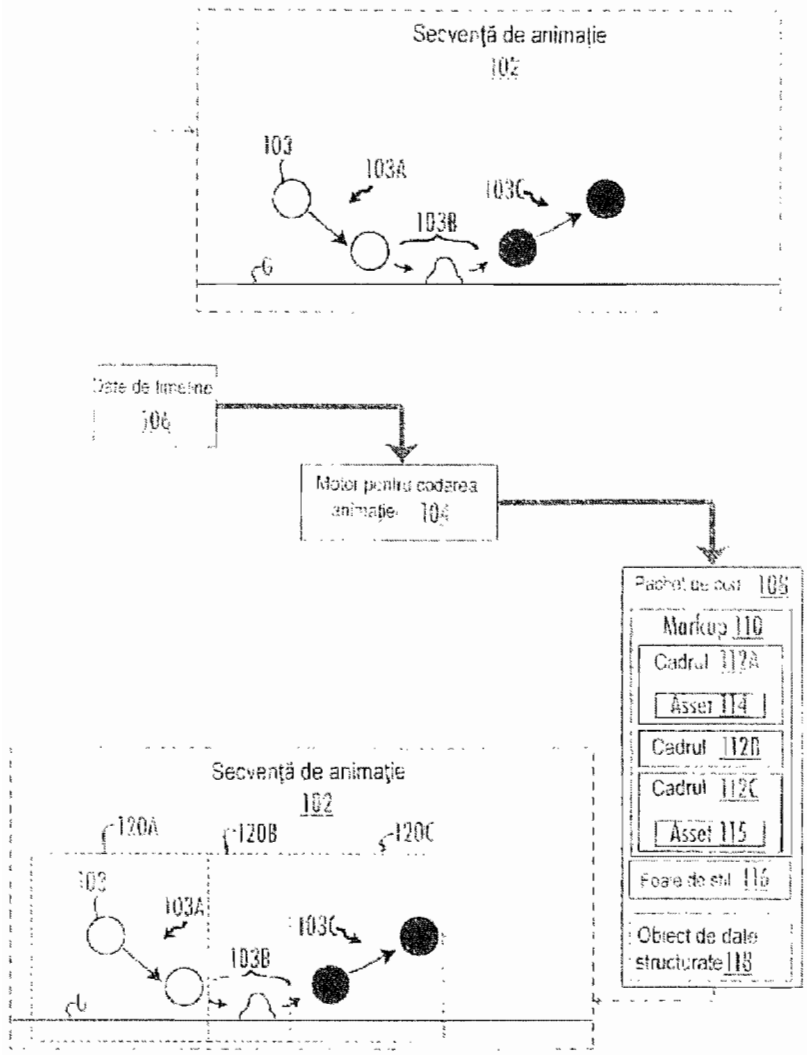

Fig. 1

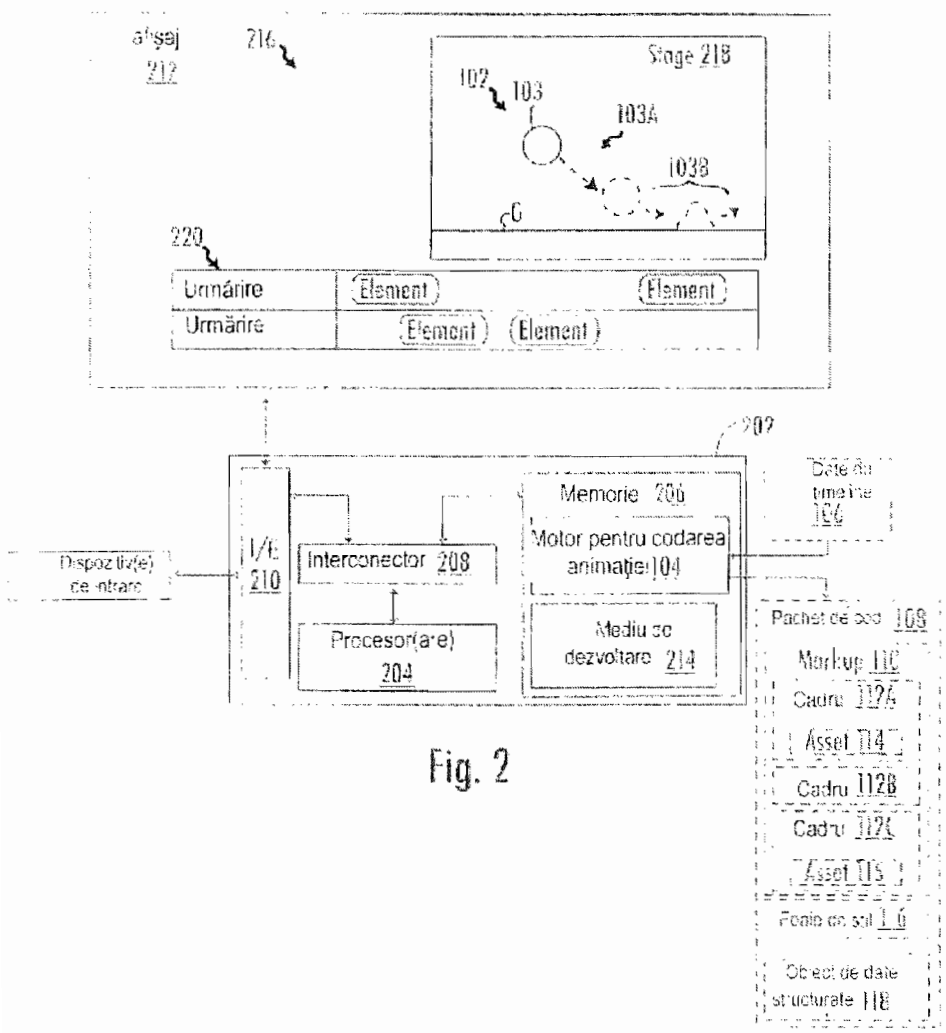
afişaj
212
216
Stage 218
102
103
103A
103B
C
220
Urmărire
Element
Element
Urmărire
Element
Element
202
Memorie 206
Motor pentru codarea animaţiei 104
Mediu de dezvoltare 214
I/E 210
Interconector 208
Procesor(are) 204
Dispoz tiv(e) de intrare
Pachet de sol 108
Markup 110
Cadru 112A
Asset 114
Cadru 112B
Cadru 112C
Asset 116
Obiect de date structurate 118

Fig. 2

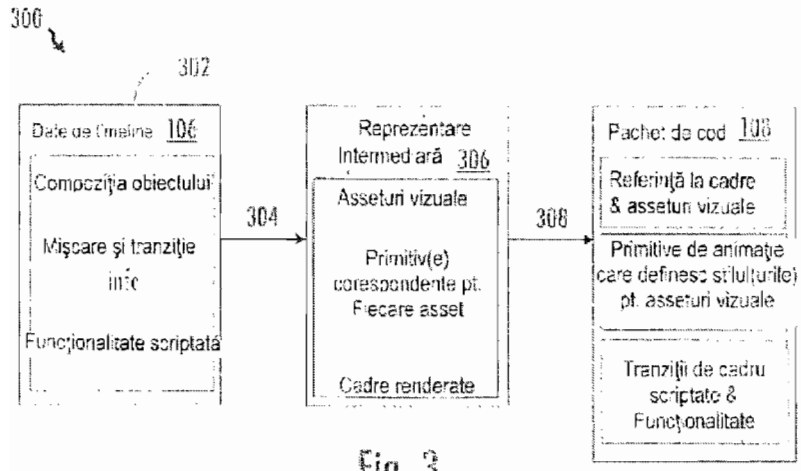
300
302
Date de Omaine 106
Compoziţia obiectului
Mişcare şi tranziţie intre
Funcţionalitate scriptată
304
Reprezentare Intermediară 306
Asseturi vizuale
Primitiv(e) corespondente pt. Fiecare asset
Cadre renderate
308
Pachet de cod 108
Referinţă la cadre & asseturi vizuale
Primitive de animaţie care definesc stilul(urile) pt. asseturi vizuale
Tranziţii de cadru scriptate & Funcţionalitate

Fig. 3

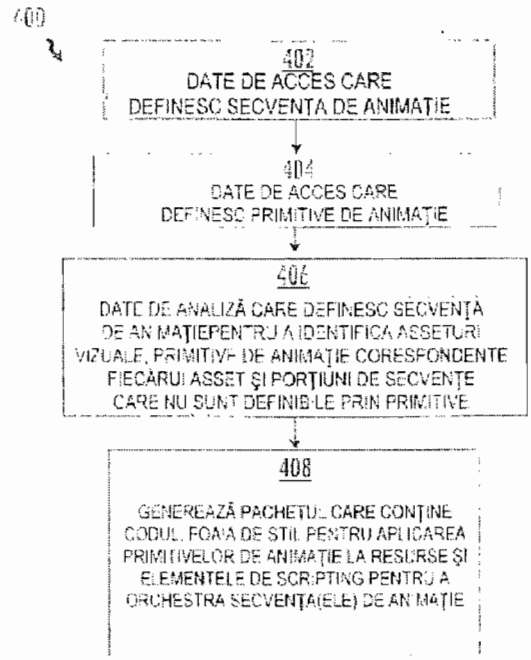
400
402
DATE DE ACCES CARE DEFINESC SECVENŢA DE ANIMAŢIE
404
DATE DE ACCES CARE DEFINESC PRIMITIVE DE ANIMAŢIE
406
DATE DE ANALIZĂ CARE DEFINESC SECVENŢĂ DE ANIMAŢIEPENTRU A IDENTIFICA ASSETURI VIZUALE, PRIMITIVE DE ANIMAŢIE CORESPONDENTE FIECĂRUI ASSET ŞI PORŢIUNI DE SECVENŢE CARE NU SUNT DEFINIBILE PRIN PRIMITIVE
408
GENEREAZĂ PACHETUL CARE CONŢINE CODUL FOAIA DE STIL PENTRU APLICAREA PRIMITIVELOR DE ANIMAŢIE LA RESURSE ŞI ELEMENTELE DE SCRIPTING PENTRU A ORCHESTRA SECVENŢA(ELE) DE ANIMAŢIE

Fig. 4

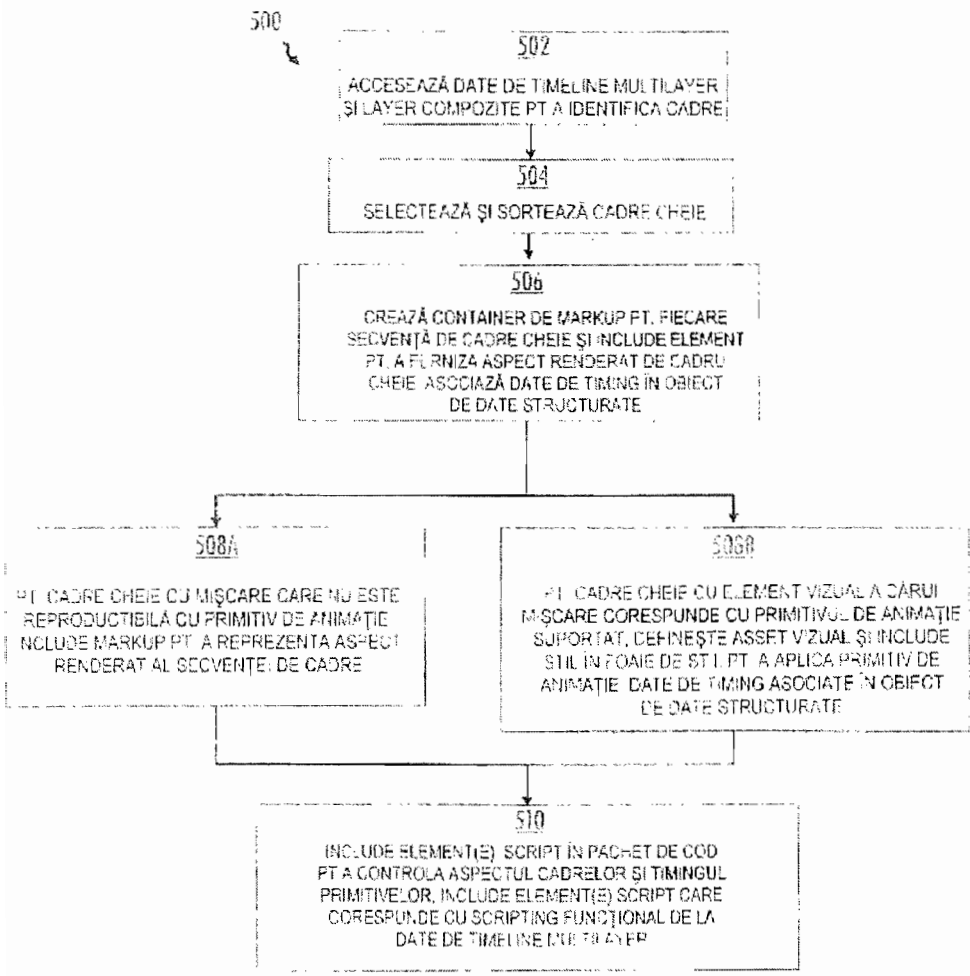
500
502
ACCESEAZĂ DATE DE TIMELINE MULTILAYER ȘI LAYER COMPOZITE PT A IDENTIFICA CADRE
504
SELECTEAZĂ ȘI SORTEAZĂ CADRE CHEIE
506
CREAZĂ CONTAINER DE MARKUP PT. FIECARE SECVENȚĂ DE CADRE CHEIE ȘI INCLUDE ELEMENT PT. A FURNIZA ASPECT RENDERAT DE CADRU CHEIE ASOCIAZĂ DATE DE TIMING ÎN OBIECT DE DATE STRUCTURATE
508A
PT CADRE CHEIE CU MIȘCARE CARE NU ESTE REPRODUCTIBILĂ CU PRIMITIV DE ANIMAȚIE INCLUDE MARKUP PT A REPREZENTA ASPECT RENDERAT AL SECVENȚEI DE CADRE
508B
PT CADRE CHEIE CU ELEMENT VIZUAL A CĂRUI MIȘCARE CORESPUNDE CU PRIMITIVUL DE ANIMAȚIE SUPORTAT, DEFINEȘTE ASSET VIZUAL ȘI INCLUDE STIL ÎN FOAIE DE STIL PT A APLICA PRIMITIV DE ANIMAȚIE DATE DE TIMING ASOCIATE ÎN OBIECT DE DATE STRUCTURATE
510
INCLUDE ELEMENT(E) SCRIPT ÎN PACHET DE COD PT A CONTROLA ASPECTUL CADRELOR ȘI TIMINGUL PRIMITIVELOR, INCLUDE ELEMENT(E) SCRIPT CARE CORESPUNDE CU SCRIPTING FUNCȚIONAL DE LA DATE DE TIMELINE DIN FLAYER

Fig. 5

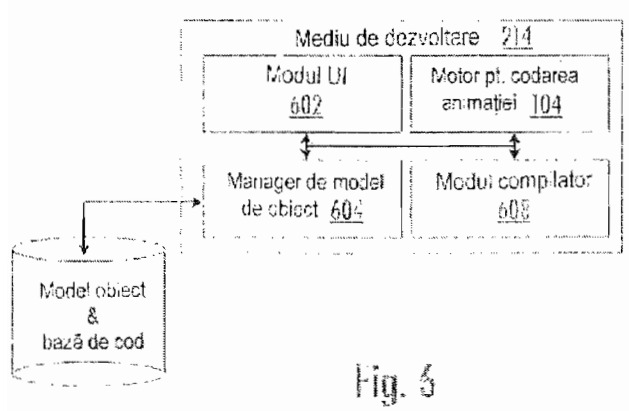
600
Mediu de dezvoltare 214
Modul UI 602
Motor pt. codarea animației 104
Manager de model de obiect 604
Modul compilator 608
Model obiect & bază de cod

Fig. 6