



(12)

CERERE DE BREVET DE INVENȚIE

(21) Nr. cerere: **a 2009 00665**

(22) Data de depozit: **28.08.2009**

(41) Data publicării cererii:
29.07.2011 BOPI nr. 7/2011

(71) Solicitant:

- **DĂSCĂLIȚA DRAGOȘ**, STR.DACIA NR.39, AP.6, IAȘI, IS, RO;
- **MĂRGELATU IONUȚ MAXIM**, STR.DOAMNA GHICA NR.1, CLĂDIREA NR.1, AP.198, BUCUREȘTI, B, RO;
- **GEORGITA DRAGOȘ**, STR.BREBU NR.1, BL.T9A, SC.A, AP.36, BUCUREȘTI, B, RO;
- **ACHIM ALIN**, STR.STAȚIEI, BL.11, SC.A, ET.1, AP.6, BUMBEȘTI-JIU, GJ, RO;
- **RAU-NEACȘU LIVIU AURELIAN**, STR.BABA NOVAC NR.23, BL.G10, SC.1, AP.31, ET.7, BUCUREȘTI, B, RO;
- **IONESCU COSTIN**, STR.NICOLAE SEBE NR.13, BUCUREȘTI, B, RO

(72) Inventatori:

- **DĂSCĂLIȚA DRAGOȘ**, STR.DACIA NR.39, AP.6, IAȘI, IS, RO;
- **MĂRGELATU IONUȚ MAXIM**, STR.DOAMNA GHICA NR.1, CLĂDIREA NR.1, AP.198, BUCUREȘTI, B, RO;
- **GEORGITA DRAGOȘ**, STR.BREBU NR.1, BL.T9A, SC.A, AP.36, BUCUREȘTI, B, RO;
- **ACHIM ALIN**, STR.STAȚIEI, BL.11, SC.A, ET.1, AP.6, BUMBEȘTI-JIU, GJ, RO;
- **RAU-NEACȘU LIVIU AURELIAN**, STR.BABA NOVAC NR.23, BL.G10, SC.1, AP.31, ET.7, BUCUREȘTI, B, RO;
- **IONESCU COSTIN**, STR.NICOLAE SEBE NR.13, BUCUREȘTI, B, RO

(74) Mandatar:

CABINET ENPORA SRL -
STR. GEORGE CĂLINESCU NR. 52A, AP.
1, SECTOR 1, BUCUREȘTI

(54) METODE ȘI SISTEM PENTRU FURNIZAREA INFORMAȚIILOR PENTRU FOLOSIRE ÎNTR-UN MEDIU CU TIMP DE RULARE OPERAȚIONAL

(57) Rezumat:

Invenția se referă la o metodă și un sistem pentru furnizarea de informații destinate a fi utilizate într-un mediu de rulare care restricționează accesul la astfel de informații. Metoda conform invenției este implementată pe calculator și cuprinde furnizarea unei aplicații runtime pe un dispozitiv de calcul, în care aplicația runtime nu poate accesa informații în cadrul mediului de rulare și utilizarea aplicației runtime pe dispozitivul de calcul prin lansarea unui executabil care preia informațiile pe care aplicația runtime nu le poate accesa din mediul runtime și face ca informațiile să devină disponibile pentru a fi utilizate de aplicația runtime, urmată de lansarea aplicației runtime care poate acum accesa informațiile puse la dispoziție de către un executabil. Sistemul conform invenției conține un dispozitiv de calcul (10) care cuprinde un mediu de rulare (14) care rulează pe un sistem de operare (13) al dispozitivului de calcul (10) și un pachet (21) care cuprinde un executabil (22) și o aplicație runtime (23), în care lansarea pachetului (21) cuprinde lansarea executabilului (22) care preia informațiile pe care aplicația runtime (23) nu le poate accesa din mediul de rulare (14) și face ca aplicațiile să devină disponibile pentru a fi utilizate de către aplicația runtime (23), urmată de lansarea aplicației runtime (23).

Revendicări: 20
Figuri: 3

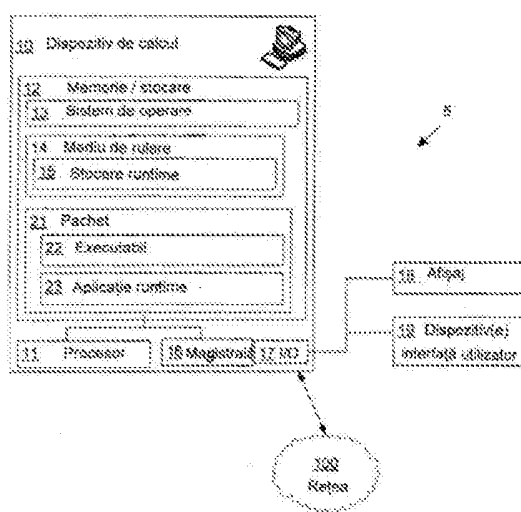
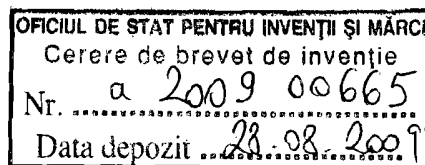


Fig. 1

Cu începere de la data publicării cererii de brevet, cererea asigură, în mod provizoriu, solicitantului, protecția conferită potrivit dispozițiilor art.32 din Legea nr.64/1991, cu excepția cazurilor în care cererea de brevet de invenție a fost respinsă, retrasă sau considerată ca fiind retrasă. Întinderea protecției conferite de cererea de brevet de invenție este determinată de revendicările conținute în cererea publicată în conformitate cu art.23 alin.(1) - (3).





DOMENIUL

[001] Această descriere, în general, se referă la software pentru calculatoare care execută, afișează, furnizează, sau folosește în alt mod conținutul electronic.

PREZENTARE GENERALĂ

[0002] Sunt disponibile multe medii de rulare (runtime environments) privind aplicațiile pentru calculatoare. Unele medii de rulare rezidă în partea superioară a sistemului de operare general al unui calculator, pentru a oferi o colecție de servicii software prin restricționarea accesului la sistemul de operare și la alte informații externe în timpul de rulare. Un mediu de rulare poate permite, de asemenea, de utilizarea aplicațiilor în cadrul timpului de rulare fără a permite acestor aplicații să acceseze anumite informații de aplicație și de sistem. Exemple de informații de sistem includ dar nu se limitează la: tipul de procesor, frecvența, memoria, spațiul disponibil pe disc, detalii privind sistemul de operare și detalii privind placa video. Exemple de informații de aplicație includ dar nu se limitează la: căile până la aplicațiile deja instalate, informații privind configurarea aplicațiilor, parametrii de inițializare, precum și alte detalii de configurare.

[0003] În timp ce există diverse beneficii în restricționarea accesului la anumite informații de către un mediu de rulare, există anumite circumstanțe în care aceste informații sunt necesare în mediul de rulare. De exemplu, aplicațiile de diagnostic care sunt utilizate pentru a evalua erorile și alte probleme pot necesita accesul la astfel de informații pentru a evalua, printre altele, dacă o eroare este rezultatul resurselor de sistem inadecvate sau al configurării eronate a sistemului. În general, poate fi de dorit ca un mediu de rulare să restricționeze în general accesul la informații în cadrul timpului de rulare, dar să permită sau să faciliteze accesul la astfel de informații în anumite alte circumstanțe. Alternativ, un mediu de rulare existent poate restricționa accesul la anumite informații și ar putea fi de dorit să se adauge capacități de acces la informații la timpul de rulare fără a schimba mediul de rulare în sine.

Rezumat

[0004] Sunt prezentate sisteme și metode pentru furnizarea de informații pentru a fi utilizate într-un mediu de rulare pentru calculator. Un executabil este lansat înainte de a lansa un mediu de rulare sau o aplicație runtime. Executabilul preia informațiile și face ca

informațiile să fie disponibile pentru utilizarea în mediul de rulare, de exemplu, prin stocarea informațiilor într-o zonă de sistem de fișiere runtime. Mediul de rulare sau aplicația runtime poate accesa apoi informațiile. De exemplu, executabilul poate obține informații de la un sistem de operare gazdă și apoi să lanseze o aplicație runtime care utilizează informațiile preluate. O aplicație care este utilizată în cadrul timpului de rulare și are nevoie de informații runtime externe poate fi furnizată ca un pachet care include un executabil și aplicația însăși. Când pachetul este lansat, executabilul preia informațiile și le pune la dispoziție pentru runtime. Runtime este apoi lansat și furnizează aplicația runtime care poate folosi aceste informații.

[0005] Un exemplu de materializare cuprinde furnizarea unei aplicații runtime pe un dispozitiv de calcul, la care aplicația runtime este utilizată în într-un mediu de rulare, ce restricționează accesul la anumite informații. Aplicația runtime este utilizată sau lansată în alt mod pe dispozitivul de calcul prin lansarea mai întâi a unui executabil care preia informațiile pe care aplicația runtime nu le poate accesa altfel, și face ca informațiile să fie disponibile pentru a fi utilizate de către aplicația runtime. Aplicația runtime este apoi lansată și poate avea acces la informațiile puse la dispoziție de către executabil. Într-un exemplu de materializare, executabilul pune la dispoziție informațiile pentru utilizarea de către aplicația runtime prin stocarea informațiilor într-un fișier într-o zonă de fișiere de sistem runtime accesibil de către aplicația runtime. Într-un alt exemplu de materializare, executabilul este pornit ca un proces și pune la dispoziție informațiile prin utilizarea unor socket-uri pentru a trece informațiile la aplicația runtime. Într-un exemplu de materializare, executabilul și aplicația runtime sunt stocate ca un singur pachet pe dispozitivul de calcul. Lansarea acestui pachet inițiază lansarea atât a executabilului cât și a aplicației runtime.

[0006] Aceste exemple de materializare nu sunt menționate pentru a limita sau defini descrierea, ci pentru a oferi exemple de materializări pentru a ajuta înțelegerea acestora. Materializările sunt discutate în Descrierea detaliată, și acolo sunt prevăzute descrieri detaliate. Avantajele oferite de diferitele materializări pot fi înțelese în continuare prin examinarea acestei specificații.

Scurtă descriere a figurilor

[0007] Aceste și alte caracteristici, aspecte, precum și avantajele prezentei comunicări sunt mai bine înțelese în cazul în care Descrierea detaliată care urmează este citită cu referire la desenele însoțitoare, în care:

Figura 1 este o schemă a sistemului care ilustrează un mediu ilustrativ de calcul; Figura 2 este o schemă de flux care ilustrează un exemplu de metodă de furnizare a informațiilor pentru a fi utilizate într-un exemplu de mediu de calcul runtime; și Figura 3 este o schemă care ilustrează un alt exemplu de metodă de furnizare a informațiilor pentru a fi utilizate într-un exemplu de mediu de calcul runtime.

Descrierea detaliată

[0008] Sunt prezentate sisteme și metode pentru furnizarea de informații pentru a fi utilizate într-un mediu de rulare. De exemplu, un mediu de rulare poate furniza o colecție de servicii de software care sunt disponibile în timpul utilizării sale, dar restricționează accesul la sistemul de operare și la alte informații. Un mediu de rulare poate permite, de asemenea, utilizarea aplicațiilor în cadrul runtime, dar poate restricționa accesul la anumite informații. Exemple de informații de sistem includ dar nu se limitează la: viteza și tipul procesorului, frecvența, memoria fizică, memoria RAM, spațiul disponibil pe disc, versiunea și detalii privind sistemul de operare, tipul și detalii privind placa video și/sau volumul sistemului de operare etc. Exemple de informații de aplicație includ dar nu se limitează la: căile până la aplicațiile deja instalate, informații privind configurarea aplicațiilor, parametrii de inițializare, precum și detalii de configurare.

[0009] Una dintre materializări prevede un executabil care este lansat înainte de a lansa un mediu de rulare. Executabilul preia informațiile și face ca informațiile să fie disponibile pentru utilizarea în mediul de rulare. Apoi, mediul de rulare se lansează și accesează informațiile. De exemplu, executabilul poate obține informații de la un sistem de operare gazdă și apoi să lanseze aplicația runtime. În una dintre materializări, aplicație care este utilizată în cadrul timpului de rulare și are nevoie de informații runtime externe este furnizată ca un pachet care include atât un executabil cât și aplicația însăși. Când pachetul este lansat, executabilul preia informațiile și le pune la dispoziție pentru runtime. Runtime este apoi lansat și furnizează aplicația runtime în mediul de rulare.

[0010] În general, informațiile pot fi puse la dispoziția unui runtime sau unei aplicații runtime într-o varietate de moduri. De exemplu, aceste informații pot fi stocate într-un fișier text, o bază de date, sau într-o locație de memorie care este furnizată sau pusă la dispoziția runtime. De exemplu, dacă un runtime are propriul sistem de fișiere sau o altă zonă de stocare, un fișier care conține informațiile preluate ar putea fi stocat în această zonă de stocare și apoi accesat din interiorul runtime. Ca un alt exemplu, executabilul care

preia informațiile ar putea fi a pornit ca un proces și apoi să utilizeze socket-uri pentru a comunica cu aplicația runtime pentru a transmite informațiile preluate.

[0011] Aceste exemple ilustrative sunt date pentru a-l introduce pe cititor în subiectul general de subiect discutat în acest document și nu sunt destinate să limiteze domeniul de aplicare a conceptelor descrise. Secțiunile care urmează descriu diferite materializări și exemple suplimentare.

Mediul de calcul ilustrativ

[0012] Referindu-se acum la desene în care, ca cifre indica ca elemente de pe întreg teritoriul figurile de mai multe, figura 1 este un sistem de diagramă care ilustrează un mediu de calcul ilustrative 5 în funcție de anumite materializări. Materializări Alte pot fi utilizate. Mediu de calcul 5 cuprinde un dispozitiv de calcul 10, că este conectat la o rețea fără fir cu fir sau 100. Aplicații exemplară care executa pe dispozitivul de calcul 10 sunt prezentate ca funcționale sau a componentelor de stocare care au reședința în memorie 12. De memorie 12 pot fi tranzitorii sau persistente. După cum este cunoscut la una de calificare în arta, astfel de aplicații pot fi rezident în orice computer adecvate de mediu ușor de citit și executa pe orice procesor adecvat. De exemplu, dispozitivul de calcul 10 mai cuprinde un calculator poate fi citit-mediul, cum ar fi o memorie cu acces aleator (RAM), cuplat la un procesor de 11 de computer care execută instrucțiunile de program executabil și / sau accesează informațiile stocate în memoria 12. O astfel de procesoare poate cuprinde un microprocesor, un ASIC, o mașină de stat, sau de alte procesor, și poate fi oricare dintre un număr de procesoare de calculator. Aceste procesoare cuprind, sau pot fi într-un mediu de comunicare cu un mediu citibil de către calculator care stochează instrucțiunile care, atunci când sunt executate de procesor, determină procesorul să efectueze pașii descriși aici.

[0013] Un mediu citibil de către calculator poate cuprinde, dar nu se limitează la un dispozitiv de stocare electronic, optic, magnetic sau un dispozitiv de stocare de alt tip. capabil de a furniza unui procesor instrucțiuni citibile de către calculator. Alte exemple cuprind, dar nu sunt limitate la dischete, CD-ROM, DVD, discuri magnetice, cipuri de memorie, ROM, RAM, un ASIC, un procesor configurat, benzi magnetice sau alte dispozitive de stocare magnetică sau alte medii de pe care procesorul unui calculator poate citi instrucțiunile. Instrucțiunile pot include instrucțiuni specifice procesorului, generate de un compilator și / sau de un interpretor de cod, scrise în orice limbă adecvată de

programare pentru calculator, inclusiv, de exemplu, C, C ++, C #, Visual Basic, Java, Python, Perl, JavaScript și ActionScript.

[0014] Rețeaua 10 prezentată cuprinde Internetul. În alte materializări, pot fi utilizate alte rețele, intraneturi, combinații de rețele, sau nici o rețea. Dispozitivul de calcul 10 poate cuprinde, de asemenea, un număr de dispozitive externe sau interne, cum ar fi un mouse, un CD-ROM, DVD, o tastatură, un afișaj, boxe audio, sau alte dispozitive de intrare sau ieșire. De exemplu, dispozitivul de calcul include 10 conexiunile 17 de intrare / ieșire, care leagă un afișaj 18 și diverse dispozitive de interfață cu utilizatorul 19. Dispozitivul de calcul 10, descris ca un sistem informatic unic, poate fi implementat ca o rețea de calculatoare, servere, sau procesoare. Exemple de dispozitive server sunt serverele, calculatoarele mainframe, calculatoarele legate în rețea, un dispozitiv bazat pe procesor, precum și tipuri similare de sisteme și dispozitive.

[0015] Un dispozitiv de calcul, cum ar fi dispozitivul exemplificativ de calcul 10, poate utiliza componente funcționale variate pentru a implementa una sau mai multe dintre caracteristicile descrise aici. Dispozitivul de calcul 10 are un sistem de operare 13 pentru operarea dispozitivului de calcul 10 și un mediu de rulare 14 pentru furnizarea de funcționalitate și aplicații runtime specifice. Aparatul exemplificativ de calcul 10 include un pachet 21, care cuprinde un executabil 22 și o aplicație runtime 23. Executabilul 22 poate fi utilizat direct din sistemul de operare 13, în timp ce aplicația runtime 23 este utilizată în runtime. Când pachetul 21 este pornit, executabilul preia orice informații necesare, cum ar fi informațiile despre dispozitivul de calcul 10, sistemul de operare 13 al acestuia, sau alte componente, sau despre orice aplicații instalate sau utilizate anterior pe dispozitivul de calcul 10. Executabilul 22 poate furniza astfel de informații pentru a fi utilizate de către aplicația runtime 23, care poate fi altfel în imposibilitatea de a accesa aceleași informații, de exemplu, din cauza limitărilor sau restricțiilor impuse de mediul de rulare 14. Într-unul dintre exemple, executabilul 22 preia sistemul de operare și alte informații și furnizează aceste informații pentru stocarea într-o stocare runtime 15 asociată cu și accesibilă din mediul de rulare 14.

[0016] Acest mediu de calcul ilustrativ 5 este oferit doar pentru a ilustra un mediu potențial care poate fi folosit pentru a implementa anumite materializări. Alte dispozitive de calcul și de configurații pot fi utilizate în mod alternativ sau suplimentar

Metode exemplificative pentru furnizarea de informații destinate utilizării într-un mediu de rulare pentru calcul

[0017] Anumite materializări furnizează informații pentru a fi utilizate într-un mediu de rulare, cum ar fi Adobe® AIR™. Adobe® AIR™ este un mediu de rulare care se află în partea superioară a unui sistem de operare și este folosit pentru a rula o serie de aplicații diverse, inclusiv aplicațiile bogate de Internet (RIA), care, de obicei, oferă experiențe mai bune din punct de vedere grafic și interactiv, decât conținutul browserelor web tradiționale. Aplicațiile Adobe® AIR™ au acces limitat la informațiile privind funcționarea sistemului și la informațiile privind setările locale. Una dintre materializările exemplificative furnizează o aplicație Adobe® AIR™ cu informații despre sistemul de operare și / sau informații cu privire la una sau mai multe aplicații Adobe® AIR™.

[0018] O varietate de tehnici pot fi folosite pentru a transmite informații către o aplicație care funcționează într-un runtime în care există un acces limitat la informațiile de sistem. Astfel de informații pot fi furnizate înainte de lansarea runtime. De exemplu, o aplicație C++, o aplicație Java, o aplicație Visual Basic, sau un alt program executabil separat poate fi utilizat pentru a prelua informații de la sistemul gazdă, înainte de lansarea unei aplicații de rulare, care va utiliza apoi informațiile. Programul executabil separat poate furniza informațiile colectate runtime-ului sau unei aplicații runtime în diferite moduri. Informațiile pot fi stocate într-o bază de date sau într-o altă zonă de memorie și / sau transmise direct aplicației. Executabilul separat ar putea transmite informațiile într-un fișier text, într-o altă zonă de memorie de stocare, sau să plaseze informațiile într-o zonă de fișiere de sistem specifice runtime care poate fi accesată de o aplicație runtime, ca exemple. Într-o materializare alternativă, executabilul separat poate fi pornit ca un proces și pot fi folosite socket-uri pentru a comunica cu runtime-ul sau cu aplicația runtime pentru a transmite informațiile.

[0019] În contextul unei aplicații de diagnosticare care rulează într-un mediu de rulare, poate fi necesar ca aplicația de diagnosticare să determine dacă dispozitivul îndeplinește cerințele minime de sistem. De exemplu, o astfel de informații poate ajuta o astfel de aplicație de diagnostic să stabilească dacă unele schimbări recente într-un sistem, ca de exemplu eliminarea unei părți din RAM, fac ca sistemul să nu mai îndeplinească anumite cerințe minime de sistem. Dacă utilizatorul nu are permisiunea de a accesa informațiile necesare, aplicația va da un mesaj.

[0020] Una dintre materializări prevede utilizarea unui executabil separat cu o aplicație de diagnostic de runtime, cum ar fi unul care rulează pe Adobe® AIR™. O astfel de aplicație de diagnostic exemplificativă poate fi o aplicație desktop stocată ca un pachet care cuprinde atât aplicația de diagnostic cât și un executabil separat. Pachetul poate fi configurat astfel încât, atunci când este lansat, lansează separat executabil înainte de lansarea runtime care rulează aplicația de diagnosticare. Executabilul separat poate obține informații pentru utilizarea ulterioară de către aplicația de diagnosticare în runtime prin furnizarea de aplicației la aplicația de diagnostic sau la o locație accesibilă prin aplicația de diagnosticare.

[0021] O aplicație exemplificativă de diagnostic utilizată într-un mediu de rulare poate fi folosită pentru a diagnostica erorile care apar în una sau mai multe alte aplicații care rulează pe calculator atât în interiorul cât și în exteriorul mediului de rulare. În consecință, informațiile furnizate unei aplicații de diagnostic se pot referi la informația la nivel general și la nivel de sistem, de exemplu, privind sistemul de operare și / sau se poate referi la informații despre o aplicație care este în curs de examinare prin aplicația de diagnosticare. Informațiile cu privire la aplicația care trebuie să fie examinată, de exemplu, poate să fi fost stocate ca un fișier jurnal pe calculator. La lansare, aplicația de diagnosticare pot fi furnizată cu sau să ofere acces la astfel de informații din fișierul jurnal de către executabilul separat. Aplicația exemplificativă de diagnostic poate utiliza informațiile din fișierul jurnal, informațiile generale sau de sistem și / sau informațiile preluate din alte surse, inclusiv, dar nu limitat la serverele externe, pentru a diagnostica și a corecta erorile.

[0022] În cazul în care un utilizator nu este în măsură de a diagnostica sau a corecta o eroare, o aplicație de diagnostic poate fi în măsură să furnizeze informațiile jurnal și / sau alte informații relevante pentru utilizator pentru a contacta o sursă externă de diagnostic, cum ar fi o linie de ajutor. De exemplu, fișierul jurnal și / sau alte informații poate fi încărcat la un server aflat la distanță, iar utilizatorul poate fi în măsură să apeleze o linie de ajutor și să discute eroarea cu un specialist care va avea acces la fișierul jurnal și la alte informații. O aplicație de diagnosticare poate transmite astfel de informații la un punct de colectare care stochează jurnalele și alte informații, inclusiv rezultatele diagnosticelor de la mai multe sisteme și utilizatori. Un punct de colectare poate permite forme statistice și alte forme de analiză care urmează să fie folosite pentru a permite unui furnizor de aplicații să înțeleagă mai bine natura și amploarea erorilor care apar în timpul utilizării aplicațiilor lor.

[0023] Figura 2 este o schemă de flux care ilustrează o metodă exemplificativă 200 de furnizare a informațiilor destinate pentru a fi utilizate într-un mediu de calcul runtime exemplificativ. Metoda exemplificativă 200 implică utilizarea unui executabil separat 210 și a unei aplicații runtime 220. De exemplu, un executabil separat și o aplicație de runtime pot fi furnizate și / sau stocate ca un pachet pe un dispozitiv de calcul, cum ar fi în pachetul exemplificativ 21 descris în figura 1.

[0024] Metoda 200 implică utilizarea aplicației runtime pornind mai întâi executabilul separat. Metoda 200 implică regăsirea informațiilor de către executabil, așa cum se arată în blocul 230. Aceste informații pot fi o parte sau toate informațiile care nu sunt disponibile aplicației runtime. Aceste informații pot include informații generale sau alte informații de sistem, cum ar fi viteza și tipul procesorului, frecvența, memoria fizică, memoria RAM, spațiul disponibil pe disc, versiunea și detaliile sistemului de operare, tipul și detaliile plăcii video și / sau volumul sistemului de operare etc. Aceste informații pot include, alternativ sau suplimentar, informații privind căile până la aplicațiile deja instalate utilizate pe dispozitivul de calcul în interiorul sau în afara timpului de rulare, parametrii de inițializare a aplicației, precum și alte detalii de configurare a aplicațiilor.

[0025] După preluarea informației folosind executabilul separat 210, metoda 200 implică lansarea aplicației runtime 220, așa cum se arată în blocul 240. Acest lucru poate implica lansarea mai întâi a mediului de rulare, dacă un astfel de mediu nu este deja lansat, și apoi lansarea aplicației runtime 220 în acel mediu. În unele materializări alternative, executabilul separat 210 și aplicația runtime 220 sunt lansate concomitent. În alte materializări alternative, este lansată mai întâi aplicația runtime 220, iar executabilul separat este lansat dacă sunt necesare informații care nu sunt accesibile din interiorul runtime-ului.

[0026] Revenind la metoda exemplificativă 200 din figura 2, după lansarea aplicației runtime 220, aplicația runtime 220 regăsește informațiile, după cum se arată în blocul 250. În acest exemplu, aplicația runtime 220 solicită informațiile de la executabilul separat 210 sau dintr-o locație de memorie asociată cu executabilul separat 210. Într-o materializare exemplificativă, executabilul separat continuă executarea până când informațiile sunt furnizate aplicației runtime 220. Într-o altă materializare exemplificativă, executabilul separat își încheie funcționarea după regăsirea informațiilor și stocarea informațiilor într-o locație accesibilă de către aplicația runtime 220.

[0027] Informațiile sunt furnizate aplicației runtime 220, așa cum se arată în blocul 260. Aceasta se poate întâmpla într-o varietate de moduri în funcție de circumstanțele speciale și de caracteristicile puse în aplicare de runtime și / sau de aplicație. Executabilul separat 210 poate fi configurat pentru a furniza în mod direct informații la cererea aplicației runtime 220. De exemplu, executabilul separat 210 ar putea fi pornit ca un proces și ar putea fi utilizate socket-uri pentru a comunica cu aplicația runtime 220 pentru a transmite informațiile.

[0028] Figura 3 este o schemă de flux care ilustrează o altă metodă exemplificativă 300 de furnizare a informațiilor pentru a fi utilizate într-un mediu de calcul runtime exemplificativ. Metoda exemplificativă 300 implică utilizarea unui executabil separat 310, a unei zone de sistem pentru fișiere runtime 320, și a unei aplicații runtime 330. Ca și în exemplul anterior din figura 2, executabilul separat 310 și o aplicație runtime 320 pot sau nu pot fi furnizate și / sau stocate ca un pachet pe un dispozitiv de calcul, cum ar fi în pachetul exemplificativ 21 descris în figura 1. Zona de sistem de fișiere runtime 320 din acest exemplu este accesibilă din interiorul mediului de rulare care rulează aplicația runtime 220. De exemplu, mediul de rulare 14 din Figura 1 ar putea fi folosit pentru a rula aplicația runtime 23, iar stocarea runtime 15 ar putea fi folosită pentru a stoca informațiile furnizate de către executabilul 22 pentru accesul de către aplicația runtime 23 și de către alte aplicații din mediul de rulare 14. Zona sistemului de fișiere runtime 320 descrisă în figura 3 poate fi localizată în dispozitivul de calcul pe care este furnizat runtime-ul sau poate fi situată în altă parte, de exemplu, într-o rețea cu locație pe Internet accesibilă din dispozitivul de calcul.

[0029] Metoda 300 poate începe într-o varietate de stadii. De exemplu, mediul de rulare folosit pentru a rula aplicația runtime 330 poate sau nu poate fi deja în curs de rulare. În general, separat față de mediul de rulare, metoda 300 începe atunci când executabilul separat 310 preia informațiile 340 pentru a fi utilizate de aplicația runtime, după cum se arată în blocul 340. Executabilul separat poate accesa informațiile privind sistemul, sistemul de operare, mediul de calcul, rețelele disponibile, dispozitivele periferice și / sau o varietate de alte tipuri de informații, în orice mijloace adecvate, cunoscute în general și utilizate de către aplicațiile care se execută printr-un sistem de operare pe o rețea sau în alt mod. Executabilul separat poate, ca un exemplu, să acceseze un registru utilizat și / sau menținut de un sistem de operare de pe dispozitivul de calcul.

[0030] După ce executabilul preia informațiile, metoda 300 presupune în continuare trimiterea de informații preluate de către executabilul separat 310 la zona sistemului de fișiere runtime 320, după cum arată blocul 350. Stocarea sau furnizarea de informații în zona sistemului de fișiere runtime 320 poate fi realizată într-o varietate de tehnici. De exemplu, informațiile pot fi compilate în unul sau mai multe fișiere noi care sunt stocate în zona sistemului de fișiere runtime 320. Alternativ, informațiile pot fi adăugate la fișierele sau jurnalele existente menținute în zona sistemului de fișiere runtime.

[0031] Metoda 200 presupune lansarea aplicației runtime 220, așa cum se arată în blocul 360. Acest lucru poate avea loc înaintea sau după blocul 350, în funcție de circumstanțe. În anumite circumstanțe, poate fi necesar ca informațiile să fie trimise în zona sistemului de fișiere runtime 320 înainte de lansarea mediului de rulare și / sau a aplicației runtime 320. În general, lansarea aplicației runtime 330 poate implica mai întâi lansarea mediului de rulare, în cazul în care un astfel de mediu nu este deja lansat, și apoi lansarea aplicației runtime 330 în acel mediu.

[0032] Odată lansată, aplicația runtime 330 poate prelua informațiile, așa cum se arată în blocul 370, iar informațiile sunt furnizate aplicației runtime, așa cum se arată în blocul 380. Informațiile pot fi preluate imediat, la lansarea aplicației, sau regăsite pe bază de necesitate. Aplicația runtime poate interacționa, de asemenea, în continuare cu zona sistemului de fișiere runtime 320 pentru a actualiza informațiile, de exemplu, prin adăugarea de informații suplimentare din cadrul mediului de rulare sau din locația accesibilă din mediul de rulare.

[0033] În general, în plus față de exemplele discutate anterior, informațiile pot fi regăsite și furnizate unui mediu de rulare sau unei aplicații runtime într-o varietate de moduri, ale căror detalii pot depinde de detaliile specifice ale mediului de rulare sau ale aplicației runtime.

Generalități

[0034] Numeroase detalii specifice sunt stabilite aici pentru a oferi o înțelegere aprofundată a subiectului revendicat. Cu toate acestea, cei calificați în domeniu vor înțelege că subiectul poate fi practicat fără aceste detalii specifice. În alte cazuri, metodele, aparatele sau sistemele care ar fi cunoscute de către cei cu cunoștințe obișnuite nu au fost descrise în detaliu, astfel încât să nu se ascundă subiectul revendicat.

[0035] Unele porțiuni sunt prezentate în termeni de algoritmi sau de reprezentări simbolice de operațiuni pe biți de date sau semnale digitale binare stocate într-o memorie a sistemului de calcul, cum ar fi memoria unui calculator. Aceste descrieri sau reprezentări algoritmice sunt exemple de tehnici utilizate de către cei cu cunoștințe obișnuite în domeniul prelucrării datelor pentru a transmite substanța muncii lor altora calificați în domeniu. Un algoritm este o secvență auto-coerentă de operațiuni de prelucrare sau similare, care să ducă la rezultatul dorit. În acest context, operațiunile sau prelucrarea implică manipularea fizică a cantităților fizice. În mod tipic, dar nu necesar, astfel de cantități pot lua forma unor semnale electrice sau magnetice care pot fi stocate, transferate, combinate, comparate sau manipulate în alt mod. S-a dovedit convenabil în timp, în principal pentru motive de utilizare comună, să se facă referință la asemenea semnale ca la biți, date, valori, elemente, simboluri, caractere, termene, numere, cifre și altele asemenea. Trebuie să se înțeleagă însă că toți acești termeni și alții asemenea trebuie să fie asociați cu cantități fizice corespunzătoare și sunt doar niște etichete convenabile. Dacă nu se afirmă în mod specific altfel, se apreciază că, în toate aceste discuții privind specificația, utilizarea unor termeni ca "procesare", "calculare", "determinare" și „identificare” sau altele asemenea se referă la acțiuni sau procese ale unei platforme de calcul, cum ar fi unul sau mai multe calculatoare și/sau un dispozitiv sau dispozitive electronice similare de calcul care manipulează sau transformă date reprezentate ca niște cantități fizice electronice sau magnetice prin memorii, registre sau alte dispozitive de stocare a informațiilor, dispozitive de transmitere sau dispozitive de afișare ale platformei de calcul.

[0036] Diferitele sisteme discutate aici nu sunt limitate la nici o arhitectură sau configurație particulară de hardware. Un dispozitiv de calcul poate include orice aranjament corespunzător de componente care furnizează un rezultat condiționat de una sau mai multe intrări. Dispozitivele adecvate de calcul includ sisteme de calculatoare multiscop bazate pe microprocesoare care accesează soft stocat care programează sau configurează sistemul de calcul de la un aparat de calcul de utilizare generală până la un aparat specializat de calcul care implementează una sau mai multe materializări ale prezentului obiect. Se poate utiliza orice limbaj sau combinație de limbaje de programare, scriptare sau de alte tipuri, pentru a implementa cunoștințele cuprinse aici într-un soft destinat pentru utilizarea în programarea sau configurarea unui dispozitiv de calcul.

[0037] Materializări ale metodelor comunicate aici pot fi realizate în exploatarea unor asemenea dispozitive de calcul. Ordinea blocurilor prezentate în exemplele de mai sus

poate varia - de exemplu, blocurile pot fi reordonate, combinate și/sau divizate în sub-blocuri. Unele blocuri sau procese pot fi realizate în paralel.

[0038] Așa cum s-a remarcat mai sus, un dispozitiv de calcul poate accesa unul sau mai multe medii citibile de calculator care încorporează în mod sensibil instrucțiuni citibile de calculator care, atunci când sunt executate de cel puțin un calculator, determină cel puțin un calculator să implementeze una sau mai multe materializări ale prezentului obiect. Când se utilizează softul, acesta poate cuprinde una sau mai multe componente, procese și/sau aplicații. Suplimentar sau alternativ la software, dispozitivul (ele) pot cuprinde circuite care fac ca dispozitivul(ele) să devină operative pentru a implementa una sau mai multe metode ale prezentului obiect.

[0039] Exemplele de dispozitive de calcul includ, dar nu sunt limitate la servere, calculatoare personale, asistenți digitali personali (PDA), telefoane celulare, televizoare, boxe set-top de televiziune și playere portabile de muzică. Dispozitivele de calcul pot fi integrate în alte dispozitive, de ex. aparate “smart”, automobile, chioșcuri și altele.

[0040] Flexibilitatea inerentă a sistemelor bazate pe calculatoare permite o mare varietate posibilă de configurații, combinații și diviziuni de sarcini și funcționalități între componente. De exemplu, procesele discutate aici pot fi implementate utilizând un singur dispozitiv de calcul sau mai multe dispozitive de calcul care să lucreze în combinație. Bazele de date și aplicațiile pot fi implementate pe un singur sistem sau distribuite pe mai multe sisteme. Componentele distribuite pot funcționa secvențial sau în paralel.

[0041] Când datele sunt obținute sau accesate între un prim sistem și un al doilea sistem de calculatoare sau de componente ale acestora, datele actuale pot circula direct sau indirect între sisteme. De exemplu, dacă un prim calculator accesează date dintr-un al doilea calculator, accesul poate implica unul sau mai multe calculatoare, proxy-uri etc. intermediare. Datele efective pot circula între primul și al doilea calculator, sau primul calculator poate furniza un cursor sau un metafișier pe care cel de-al doilea calculator îl utilizează pentru a accesa datele efective dintr-un alt calculator decât primul calculator, de exemplu. Datele pot fi “trase” printr-o cerere sau “împinse” fără o cerere, în diferite materializări..

[0042] Tehnologia descrisă aici face, de asemenea, referință la comunicarea datelor între componente sau sisteme. Se va aprecia că o asemenea comunicare poate avea loc prin orice număr sau tip adecvat de rețele sau linkuri, incluzând dar nefiind limitată la rețele comutate, o rețea locală (LAN), o rețea lărgită (WAN), rețeaua publică de telefonie

comutată (PSTN), Internetul, un intranet sau orice combinație de linkuri de comunicare cu sau fără fir.

[0043] Orice mediu tangibil adecvat citibil de către calculator poate fi utilizat pentru implementarea sau practicarea obiectului comunicat prin prezenta, inclusiv dar nefiind limitat la dischete, unități de disc, medii de stocare magnetică, medii de stocare optică, inclusiv discurile (incluzând CD-ROM, DVD-ROM și variante ale acestora), memorii flash, RAM, ROM, și alte dispozitive de memorie.

[0044] Utilizarea aici a expresiilor “adaptat pentru” sau “configurat pentru” este înțeleasă ca un limbaj deschis și inclusiv care nu rezolvă dispozitive adaptate sau configurate pentru a efectua sarcini sau pași suplimentari. În plus, utilizarea expresiei “bazat pe” este înțeleasă a fi deschisă și inclusivă, în sensul că, în practică, un proces, un pas, un calcul sau o altă acțiune “bazată pe” una sau mai multe condiții sau valori enumerate poate să fie bazată, în practică, pe condiții sau valori suplimentare peste cele enumerate. Anteturile, listele și numerotările incluse aici servesc numai pentru înlesnirea explicării și nu sunt înțelese a fi limitative.

[0045] Deși prezentul obiect a fost descris în detaliu cu referire la materializări specifice ale acestuia, se va aprecia că cei cu cunoștințe în domeniu, după înțelegerea celor de mai sus, pot realiza cu ușurință modificări, variante și echivalente ale unor asemenea materializări. În mod corespunzător, se va înțelege că prezenta comunicare a fost prezentată mai mult pentru exemplificare decât ca limitare și nu împiedică includerea unor modificări, variații și/sau adăugări la prezentul obiect, așa cum va fi evident, cu ușurință, pentru o persoană cu cunoștințe obișnuite în acest domeniu.

REVENDICĂRI

1. O metodă implementată pe calculator, cuprinzând:
furnizarea unei aplicații runtime pe un dispozitiv de calcul, aplicația runtime utilizată în cadrul unui mediu de rulare pe dispozitivul de calcul, în care aplicația runtime nu poate accesa informațiile în cadrul mediului de rulare;
utilizarea aplicației runtime pe dispozitivul de calcul prin:
lansarea unui executabil pe un dispozitiv de calcul, în care executabilul preia informațiile pe care aplicația runtime nu le poate accesa din mediul runtime și face ca informațiile să fie disponibile pentru a fi utilizate de aplicația runtime; precum și lansarea aplicației runtime, în care aplicația runtime poate accesa informațiile puse la dispoziție de către executabil.
2. Metoda conform revendicării 1, în care executabilul face ca informațiile să fie disponibile pentru utilizarea de către aplicația runtime prin stocarea informațiilor într-o zonă a sistemului de fișiere runtime accesibilă de către aplicația runtime.
3. Metoda conform revendicării 1, în care executabilul este pornit ca un proces, în care executabilul face ca informațiile să fie disponibile pentru utilizarea de către aplicația runtime folosind socket-uri pentru a transmite informațiile către aplicație runtime.
4. Metoda conform revendicării 1, în care executabilul și aplicația runtime sunt stocate ca un singur obiect pe dispozitivul de calcul și în care lansarea obiectului unic inițiază lansarea atât a executabilului cât și a aplicației runtime.
5. Metoda conform revendicării 1, în care informațiile specifică: viteza procesorului, tipul procesorului, mărimea memoriei fizice, mărimea memoriei RAM, spațiul disponibil pe disc, versiunea sistemului de operare sau tipul plăcii video.
6. Metoda conform revendicării 1, în care informațiile specifică o cale până la o altă aplicație pe dispozitivul de calcul.

7. Metoda conform revendicării 1, în care lansarea executabilului are loc înainte de lansarea aplicației runtime.
8. Metoda conform revendicării 1, în care lansarea executabilului are loc în timp ce mediul de rulare nu este în execuție.
9. Metoda conform revendicării 1, în care mediul de rulare este Adobe AIR care rulează în partea superioară a unui sistem de operare de pe dispozitivul de calcul și în care informațiile reprezintă informații privind sistemul de operare.
10. Metoda conform revendicării 1, în care aplicația runtime este o aplicație de diagnostic care necesită informațiile pentru a diagnostica erori.
11. Un aparat de calcul care cuprinde:
un mediu de rulare specificat de instrucțiunile din aparatul de calcul, în care mediul de rulare care rulează pe un sistem de operare al aparatului de calcul restricționează accesul la informații și;
un pachet specificat de instrucțiunile din aparatul de calcul, în care lansarea pachetului cuprinde:
lansarea unui executabil pe un aparat de calcul, după care executabilul preia informațiile pe care o aplicație runtime nu le poate accesa din mediul de rulare și face ca informațiile să fie disponibile pentru utilizarea de către aplicația runtime, și
lansarea aplicației runtime, după care aplicația runtime poate accesa informațiile puse la dispoziție de executabil.
12. Aparat de calcul conform revendicării 11, în care lansarea pachetului cuprinde mai departe rularea mediului de rulare după lansarea executabilului.
13. Aparat de calcul conform revendicării 11, în care executabilul face ca informațiile să fie disponibile pentru utilizarea de către aplicația runtime prin stocarea informațiilor într-o zonă a sistemului de fișiere runtime accesibilă de către aplicația runtime.

14. Aparat de calcul conform revendicării 11, în care executabilul este pornit ca un proces, la care executabilul face ca informațiile să fie disponibile pentru utilizarea de către aplicația de runtime folosind socket-uri pentru a transmite informațiile la aplicația runtime.

15. Aparat de calcul conform revendicării 11, în care informațiile specifică: viteza procesorului, tipul procesorului, mărimea memoriei fizice, mărimea memoriei RAM, spațiul disponibil pe disc, versiunea sistemului de operare sau tipul plăcii video.

16. Aparat de calcul conform revendicării 11, în care mediul de rulare este Adobe AIR și în care informațiile reprezintă informații privind sistemul de operare.

17. Aparat de calcul conform revendicării 11, în care aplicația runtime este o aplicație de diagnostic care necesită informațiile pentru a diagnostica erori.

18. Un mediu citibil de către calculator pe care este codat un cod program, codul program cuprinzând:

cod program pentru furnizarea unei aplicații runtime pe un dispozitiv de calcul, aplicația runtime fiind utilizată în cadrul unui mediu de rulare pe dispozitivul de calcul, în care aplicația runtime nu poate accesa informațiile în mediul de rulare;

cod program pentru lansarea aplicației runtime pe dispozitivul de calcul prin:

lansarea unui executabil pe un dispozitiv de calcul, în care executabilul preia informațiile pe care aplicația runtime nu le poate accesa din mediul runtime și face ca informațiile să fie disponibile pentru utilizarea de către aplicația runtime; precum și

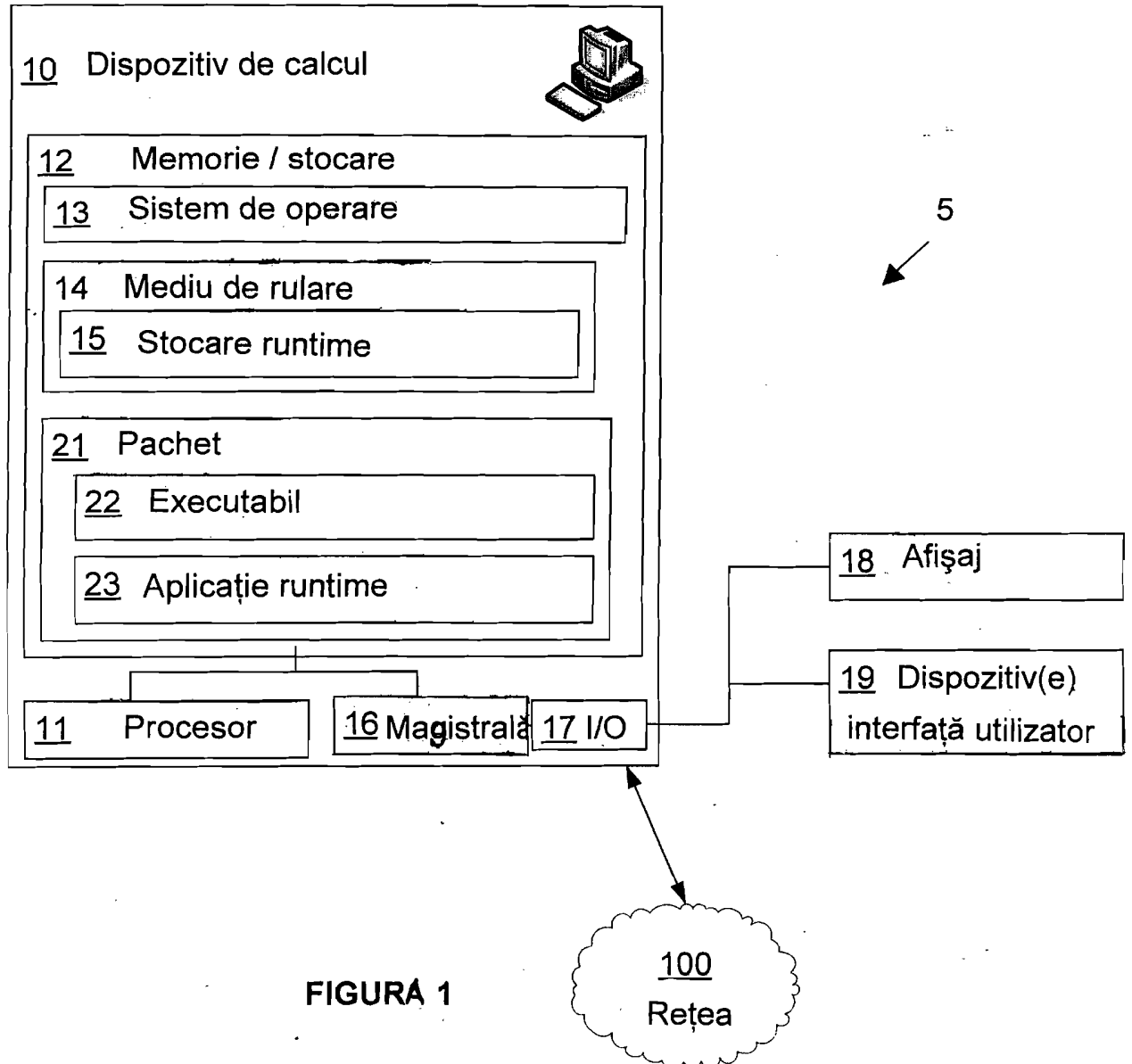
lansarea aplicației runtime, după care aplicația runtime poate accesa informațiile puse la dispoziție de către executabil.

19. Mediu citibil de către calculator conform revendicării 18, în care executabilul face ca informațiile să fie disponibile pentru utilizarea de către aplicația runtime prin stocarea informațiilor într-o zonă a sistemului de fișiere runtime accesibilă de către aplicația runtime.

20. Mediu citibil de către calculator conform revendicării 18, în care executabilul este pornit ca un proces, în care executabilul face ca informațiile să fie disponibile pentru

utilizarea de către aplicația de runtime folosind socket-uri pentru a transmite informațiile la aplicația runtime.

=====



200

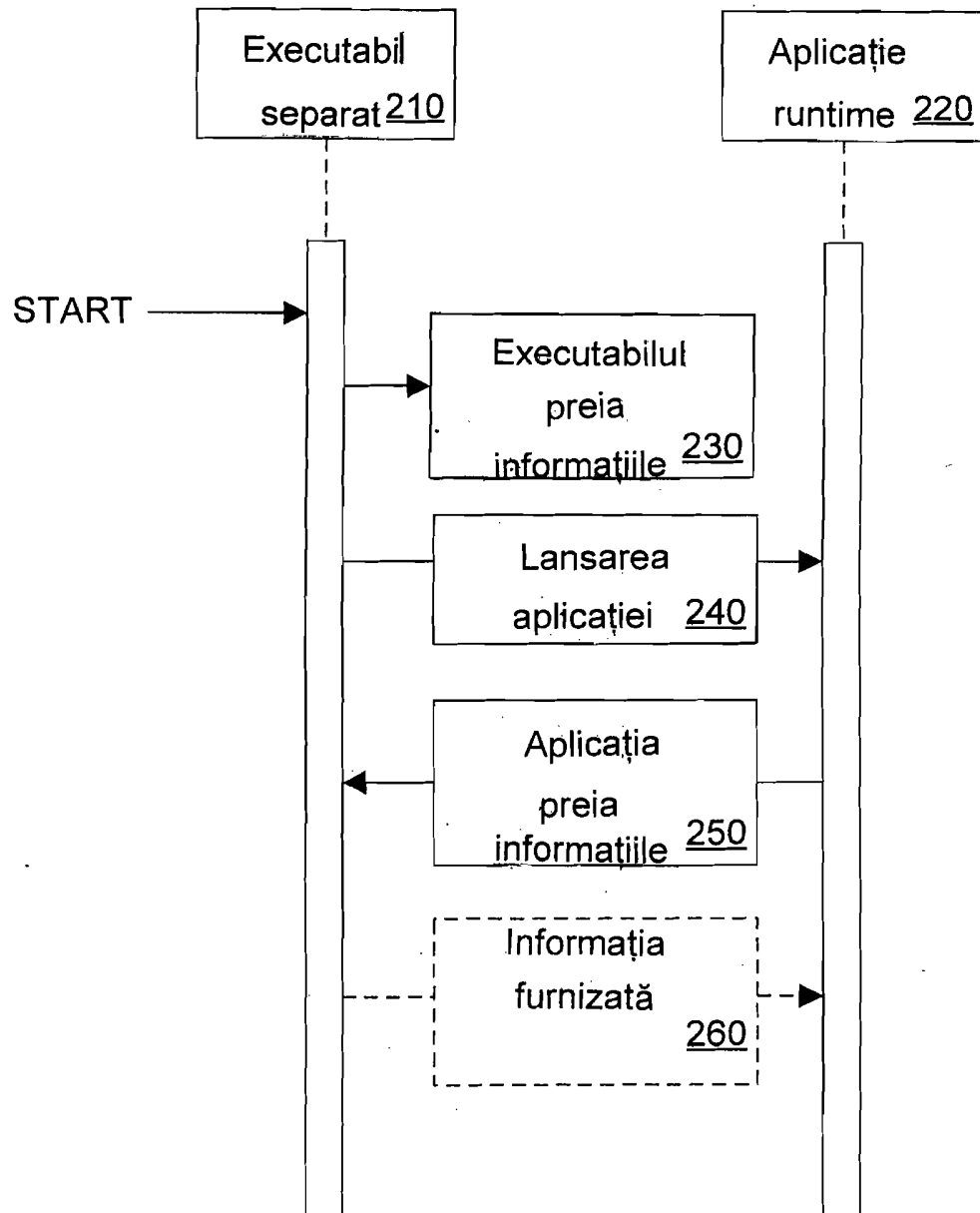


FIGURA 2

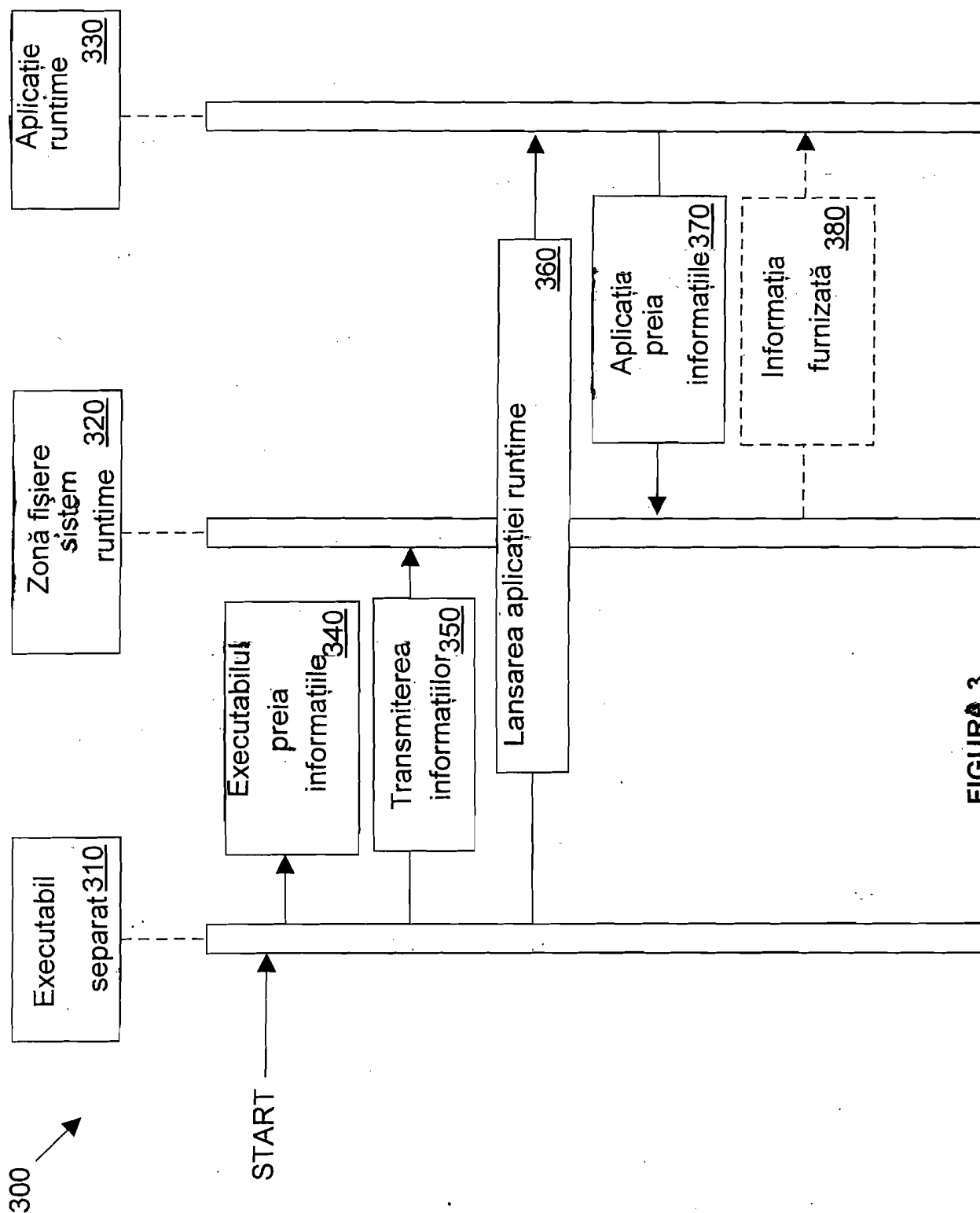


FIGURA 3