(12) DEMANDE INTERNATIONALE PUBLIÉE EN VERTU DU TRAITÉ DE COOPÉRATION EN MATIÈRE DE BREVETS (PCT)

(19) Organisation Mondiale de la Propriété Intellectuelle
Bureau international

(43) Date de la publication internationale
22 août 2013 (22.08.2013)

WIPO | PCT

(10) Numéro de publication internationale
WO 2013/121148 A1

(51) Classification internationale des brevets :
G06F 12/10 (2006.01) *G06F 11/36* (2006.01)

(21) Numéro de la demande internationale :
PCT/FR2013/050297

(22) Date de dépôt international :
14 février 2013 (14.02.2013)

(25) Langue de dépôt : français

(26) Langue de publication : français

(30) Données relatives à la priorité :
1251404 15 février 2012 (15.02.2012) FR

(71) Déposant : **BULL SAS** [FR/FR]; Rue Jean Jaurès, F-78340 Les Clayes Sous Bois (FR).

(72) Inventeurs : **BRU, Xavier**; 10 C, rue Fantin Latour, F-38640 Claix (FR). **DERR, Simon**; 3, rue de la Gaité, F-38600 Fontaine (FR). **MENYHART, Zoltan**; 109, Avenue Jean Perrot, F-38100 Grenoble (FR).

(74) Mandataire : **SANTARELLI**; BP 237, 14 avenue de la Grande Armée, F-75822 Paris Cedex 17 (FR).

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

(84) États désignés *(sauf indication contraire, pour tout titre de protection régionale disponible)* : ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), eurasien (AM, AZ, BY, KG, KZ, RU, TJ, TM), européen (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

[Suite sur la page suivante]

(54) Title : METHOD AND DEVICE FOR MANAGING THE EXECUTION OF PROCESSES IN A COMPUTER SYSTEM

(54) Titre : PROCÉDÉ ET DISPOSITIF DE GESTION D'EXÉCUTION DE PROCESSUS DANS UN SYSTÈME INFORMATIQUE

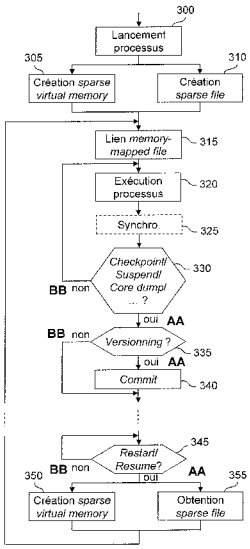


Fig. 3

300 Process initiation
305 Creation of sparse virtual memory
310 Creation of sparse file
315 Memory-mapped file relation
320 Process execution
325 Synchronisation
350 Creation of sparse virtual memory
355 Obtaining of sparse file
AA oui
BB no

(57) Abstract : The invention concerns, in particular, managing the execution of computer processes in a computer comprising a working memory and a mass memory and using an operating system configured to manage the virtual working memory. When a process is initiated, a region of sparse virtual memory pertaining to said initiated process and a sparse file associated with said initiated process are created (305, 310). A correspondence relation is then established (315) between the region of sparse virtual memory and the sparse file which have been created, according to a management function of memory-mapped files.

(57) Abrégé : L'invention a notamment pour objet la gestion d'exécution de processus informatiques dans un ordinateur comprenant une mémoire de travail et une mémoire de masse et utilisant un système d'exploitation configuré pour gérer de la mémoire de travail virtuelle. Lors du lancement d'un processus, une région de mémoire virtuelle éparse propre audit processus lancé et un fichier éparse associé au processus lancé sont créés (305, 310). Un lien de correspondance est ensuite établi (315) entre la région de mémoire virtuelle éparse et le fichier éparse créés, selon une fonction de gestion de fichiers cartographiés en mémoire.

WO 2013/121148 A1

Déclarations en vertu de la règle 4.17 :

— *relative à la qualité d'inventeur (règle 4.17.iv))*

Publiée :

— *avec rapport de recherche internationale (Art. 21(3))*

Procédé et dispositif de gestion d'exécution de processus dans un système informatique

La présente invention concerne l'exécution de code informatique et plus particulièrement un procédé et un dispositif de gestion d'exécution de processus dans un système informatique.

Afin de faciliter la programmation des systèmes informatiques, des couches logicielles intermédiaires sont généralement utilisées entre la couche matérielle et la couche logicielle applicative. De telles couches intermédiaires permettent d'exécuter des tâches aussi génériques que possible, telles que des transferts et des traitements de données. En outre, l'utilisation de telles tâches permet souvent de décorréler les couches applicative et matérielle, autorisant ainsi une application logicielle à être exécutée par plusieurs couches matérielles différentes.

Ces couches intermédiaires peuvent également être utilisées pour des fonctions particulières, notamment des fonctions de mémorisation d'états de mémoire de travail (ou mémoire vive) d'un système informatique lors de l'arrêt et la reprise d'exécution de processus informatiques. Ces fonctions sont mises en œuvre, en particulier, pour le développement et la mise au point de ces processus.

Ainsi, par exemple, il existe des fonctions d'interruption et de reprise de processus, appelées *checkpoint* et *restart* en terminologie anglo-saxonne, pour suspendre l'exécution d'un processus à un point donné de celui-ci et le reprendre à ce point ultérieurement. De telles fonctions sont typiquement utilisées pour sauvegarder des données liées à l'exécution du processus considéré au cours de son exécution lorsque celle-ci est potentiellement longue de telle sorte qu'il ne soit pas nécessaire de ré-exécuter tout le processus si une erreur matérielle ou logicielle survient au cours de son exécution ou lorsqu'une ressource nécessaire à son exécution s'avère indisponible. En d'autres termes, ces fonctions permettent de sauvegarder l'état exact de

processus informatiques et, en cas de pannes, restaurer l'état exact des processus et les redémarrer à partir du point de sauvegarde, comme si rien n'était arrivé.

La mise en œuvre d'une fonction de type *checkpoint* consiste à recopier l'intégralité des données des processus vers une mémoire de masse (périphériques de stockage) lorsque la fonction est appelée.

Cependant, lorsque le système informatique exécutant le processus utilise de la mémoire virtuelle, il est généralement nécessaire d'accéder à des données présentes dans la mémoire de masse avant de les recopier vers celle-ci. En effet, la mémoire virtuelle consiste à utiliser une partie de la mémoire de masse comme mémoire de travail en utilisant un mécanisme d'échange (appelé *swap* en terminologie anglo-saxonne) pour charger en mémoire de travail réelle des données présentes en mémoire de masse lorsque des opérations font appel à celles-ci. Ainsi, lorsque beaucoup de données sont stockées en mémoire de masse du fait de l'utilisation de mémoire virtuelle, beaucoup d'opérations d'entrée/sortie additionnelles sont nécessaires pour accéder à ces données.

De façon similaire, il existe des fonctions de suspension d'exécution de processus et de reprise d'exécution, appelées *suspend* et *resume* en terminologie anglo-saxonne, qui ont pour objet d'arrêter temporairement l'exécution d'un processus à un instant donné et permettre sa reprise ultérieurement au point où il a été arrêté, à l'aide de données mémorisées lors de l'arrêt. Ces fonctions sont généralement utilisées pour mettre temporairement fin à l'exécution d'un processus, notamment lorsque plusieurs processus doivent être exécutés par un système informatique et qu'il existe des niveaux de priorité associés à ces processus. En d'autres termes, ces fonctions permettent de suspendre temporairement l'exécution de processus informatiques et laisser d'autres processus profiter des ressources de l'équipement informatique puis restaurer les processus suspendus et les faire redémarrer comme s'ils continuaient à partir du point de suspension, comme si rien n'était arrivé.

De façon similaire encore, il existe une fonction de sauvegarde de mémoire de travail, appelée *core dump* en terminologie anglo-saxonne, qui consiste à mémoriser l'état d'une mémoire de travail à un instant donné, typiquement lorsqu'un processus se termine de façon anormale. Cette fonction est généralement utilisée pour analyser a posteriori l'exécution d'un processus et chercher à comprendre les raisons ayant conduit son arrêt anormal.

Ces fonctions peuvent être utilisées de façon autonome ou combinées à d'autres dans des outils de mise au point de processus informatiques permettant généralement une exécution pas-à-pas d'un processus.

Ces fonctions étant particulièrement importantes pour l'exécution de processus dans des systèmes informatiques, il existe un besoin constant de les améliorer, notamment en termes de performances, et de les simplifier.

L'invention a ainsi pour objet un procédé de gestion d'exécution d'au moins un processus dans un ordinateur comprenant une mémoire de travail et une mémoire de masse, ledit ordinateur comprenant un système d'exploitation configuré pour gérer de la mémoire de travail virtuelle, ce procédé comprenant les étapes suivantes lors du lancement dudit au moins un processus,

- création d'au moins une région de mémoire virtuelle éparse propre audit au moins un processus lancé ;
- création d'au moins un fichier éparse associé audit au moins un processus lancé ; et,
- établissement d'un lien de correspondance entre ladite au moins une région de mémoire virtuelle éparse et ledit au moins un fichier éparse créés, selon une fonction de gestion de fichiers cartographiés en mémoire.

L'invention permet de gérer le code exécutable de processus et les données créées et modifiées par ces processus pour des fonctions d'interruption et de reprise, de suspension d'exécution et de sauvegarde de la mémoire de travail du fait que

- les données d'un processus évoluent virtuellement dans le fichier éparse correspondant (une certaine partie des données ayant déjà été écrite dans le fichier correspondant il y a par conséquent, statistiquement, moins

d'opérations d'entrée/sortie à effectuer lors d'appels à des fonctions d'interruption, de suspension d'exécution et de reprise qu'une solution standard) ;

- il n'est pas nécessaire d'aller chercher une partie des données sur un périphérique en utilisant un mécanisme d'échange (*swap*) ;

- un nouvel espace de mémoire virtuelle peut être créé au moment de la restauration d'un processus, éventuellement défaillant, en y associant le fichier éparse correspondant par un mécanisme de gestion de fichiers cartographiés en mémoire.

Selon un mode de réalisation particulier, le procédé comprend en outre une étape d'exécution dudit au moins un processus, selon lequel, durant ladite étape d'exécution dudit au moins un processus, ledit système d'exploitation copie des données associées audit au moins un processus et mémorisées dans ladite au moins une région de mémoire virtuelle éparse créée vers ledit au moins un fichier éparse créé.

Toujours selon un mode de réalisation particulier, le procédé comprend en outre, suite à la validation d'une modification d'une donnée associée audit au moins un processus et mémorisée dans ladite au moins une région de mémoire virtuelle éparse créée et recopiée dans ledit au moins un fichier éparse créé, une étape de création d'une nouvelle version dudit au moins un fichier éparse créé.

Le procédé selon l'invention permet ainsi de conserver un historique d'évolution du processus et de n'utiliser des blocs mémoires des périphériques de stockage que pour les données modifiées.

Le procédé selon l'invention peut notamment être mis en œuvre pour effectuer une étape de reprise d'exécution dudit au moins un processus suite à une d'interruption d'exécution de celui-ci.

Selon un mode de réalisation particulier, ladite étape de reprise d'exécution dudit au moins un processus comprend les étapes suivantes,

- création d'au moins une région de mémoire virtuelle éparse propre audit au moins un processus relancé ;

- obtention dudit au moins un fichier éparse préalablement créé et associé audit au moins un processus lancé ; et,

- établissement d'un lien de correspondance entre ladite au moins une région de mémoire virtuelle éparse créé propre audit au moins un processus relancé et ledit au moins un fichier éparse obtenu, selon une fonction de gestion de fichiers cartographiés en mémoire.

Toujours selon un mode de réalisation particulier, au moins l'une des étapes de création d'au moins une région de mémoire virtuelle éparse, de création d'au moins un fichier éparse et d'établissement d'un lien de correspondance entre une région de mémoire virtuelle éparse et un fichier éparse est une fonction dudit système d'exploitation. Le procédé selon l'invention est ainsi particulièrement simple à mettre en œuvre et limite les développements spécifiques nécessaires à sa mise en œuvre.

Le procédé selon l'invention peut notamment être utilisé dans un outil de mise au point d'au moins un processus.

L'invention a également pour objet un programme d'ordinateur comprenant des instructions adaptées à la mise en œuvre de chacune des étapes du procédé décrit précédemment lorsque ledit programme est exécuté sur un ordinateur ainsi qu'un dispositif comprenant des moyens adaptés à la mise en œuvre de chacune des étapes du procédé décrit précédemment. Les avantages procurés par ce programme d'ordinateur et ce dispositif sont similaires à ceux évoqués précédemment.

D'autres avantages, buts et caractéristiques de la présente invention ressortent de la description détaillée qui suit, faite à titre d'exemple non limitatif, au regard des dessins annexés dans lesquels :

- la figure 1 illustre un exemple de dispositif de traitement d'informations adapté à mettre en œuvre l'invention ;

- la figure 2 illustre schématiquement un exemple d'un mécanisme de chargement de code d'un processus d'une mémoire de masse vers une mémoire de travail virtuelle d'un système informatique tel que celui décrit en référence à la figure 1 ; et,

- la figure 3 illustre certaines étapes de gestion d'exécution de processus mises en œuvre conformément à l'invention.

De façon générale, l'invention a pour objet la fourniture de fonctions de gestion d'exécution de processus dans un système informatique, ces fonctions étant basées sur des fonctions existantes, simples et efficaces, de gestion de fichiers et de gestion de mémoire de travail.

Elle permet la prise en compte des mécanismes décrits ci-avant à travers un dispositif logiciel unique consistant notamment à associer à chaque processus un fichier contenant les données modifiées du processus géré directement par le dispositif de mémoire virtuelle en remplacement de l'espace disque d'échange (ou espace disque dit de *swap*) utilisé par les systèmes d'exploitation.

Le dispositif de mémoire virtuelle permet d'établir une correspondance directe entre l'espace virtuel du processus et les blocs du fichier associé à ce processus (fichier dit mappé).

Seules les pages de mémoire virtuelle contenant des données modifiées par le processus sont susceptibles d'être réécrites par le dispositif de mémoire virtuelle dans ce fichier (fichier dit éparse).

La figure 1 illustre un exemple de dispositif de traitement d'informations pouvant être utilisé pour mettre en œuvre, au moins partiellement, l'invention. Le dispositif 100 est par exemple un micro-ordinateur de type PC (sigle de *Personal Computer* en terminologie anglo-saxonne), un terminal relié à un serveur ou un serveur.

Le dispositif 100 comporte de préférence un bus de communication 102 auquel sont reliés :

- une unité centrale de traitement ou microprocesseur 104 (CPU, sigle de *Central Processing Unit* en terminologie anglo-saxonne) ;
- une mémoire morte 106 (ROM, acronyme de *Read Only Memory* en terminologie anglo-saxonne) pouvant comporter un système d'exploitation, par exemple un système d'exploitation de type Linux (Linux est une marque) et des programmes tels que "Prog" ;

- une mémoire de travail, mémoire vive ou mémoire cache 108 (RAM, acronyme de *Random Access Memory* en terminologie anglo-saxonne) comportant des registres adaptés à enregistrer des variables et paramètres créés et modifiés au cours de l'exécution des programmes précités ; et,

- une mémoire de masse 120 telle qu'un disque dur ou une mémoire de type mémoire flash ou SSD (sigle de *Solid-State Drive* en terminologie anglo-saxonne) pouvant être utilisée pour former une mémoire de travail virtuelle et pouvant comporter les programmes "Prog" précités et des données traitées ou à traiter selon l'invention

Optionnellement, le dispositif 100 peut également disposer des éléments suivants :

- une carte graphique 114 reliée à un écran 116 ;
- un clavier 122 et une souris 124 ou tout autre dispositif de pointage comme un crayon optique, un écran tactile ou une télécommande permettant à l'utilisateur d'interagir avec le dispositif 100 ;
- une interface de communication 126 reliée à un réseau de communication distribué 128, par exemple un réseau Ethernet ou un Infiniband (Ethernet et Infiniband sont des marques), l'interface étant apte à transmettre et à recevoir des données ; et,
- un lecteur de cartes mémoires (non représenté) adapté à y lire ou à y écrire des données traitées ou à traiter.

Le bus de communication permet la communication et l'interopérabilité entre les différents éléments inclus dans le dispositif 100 ou reliés à lui. La représentation du bus n'est pas limitative et, notamment, l'unité centrale est susceptible de communiquer des instructions à tout élément du dispositif 100 directement ou par l'intermédiaire d'un autre élément du dispositif 100.

Le code exécutable de chaque fonction permettant au dispositif 100 de mettre en œuvre l'invention, peut être stocké, par exemple, dans la mémoire de masse 120 ou en mémoire morte 106. Selon une variante, le code exécutable des fonctions pourra être reçu par l'intermédiaire du réseau de communication 128, via l'interface 126, pour être stocké de façon identique à

celle décrite précédemment. De manière plus générale, la ou les fonctions pourront être chargées dans un des moyens de stockage du dispositif 100 avant d'être exécutés.

L'unité centrale 104 va commander et diriger l'exécution des instructions ou portions de code logiciel du ou des fonctions selon l'invention, instructions qui sont stockées dans la mémoire de masse 120 ou dans la mémoire morte 106 ou bien dans les autres éléments de stockage précités. Lors de la mise sous tension, le ou les programmes qui sont stockés dans une mémoire non volatile, par exemple la mémoire de masse 120 ou la mémoire morte 106, comprenant la ou les fonctions selon l'invention, typiquement, en particulier, un système d'exploitation, sont transférés dans la mémoire vive 108 qui contient alors le code exécutable de la ou des fonctions selon l'invention, ainsi que des registres pour mémoriser les variables et paramètres nécessaires à la mise en œuvre de l'invention.

Il est rappelé ici qu'il existe dans certains systèmes d'exploitation un mécanisme de gestion de mémoire virtuelle permettant de créer une mémoire virtuelle éparse, appelée *sparse virtual memory* en terminologie anglo-saxonne. Il fait croire aux processus informatiques que la taille de la mémoire de travail (mémoire vive) est très importante (plus grande de plusieurs fois l'ordre de grandeur de la taille de mémoire vive physique). Selon ce mécanisme, le système d'exploitation ne matérialise que le faible pourcentage de la mémoire vive virtuelle auquel un processus accède, la grande majorité de la mémoire vive virtuelle n'existant pas.

La mise en œuvre de ce mécanisme est typiquement basée sur un mécanisme d'échange (appelé *swap* en terminologie anglo-saxonne), selon lequel une mémoire de masse est utilisée pour notamment permettre à des processus de s'exécuter dans un environnement matériel possédant moins de mémoire de travail (mémoire vive) que nécessaire.

Il est également rappelé qu'un fichier éparse, appelé *sparse file* en terminologie anglo-saxonne, est un fichier informatique dont la taille théorique est particulièrement importante, typiquement supérieure à la capacité de la mémoire de masse utilisée pour le stocker, et qui comporte des « vides », c'est-

à-dire des emplacements qui ne sont jamais accédés. Du fait de ces derniers, il existe des mécanismes pour les stocker de façon optimisée, les emplacements jamais accédés n'étant pas mémorisés. Lors de la lecture de fichiers éparses, le système de fichiers converti les blocs représentant des « vides » en blocs réels ne comprenant que des bits nuls.

Il est également rappelé que des fichiers cartographiés en mémoire, aussi appelés *memory-mapped files* en terminologie anglo-saxonne, sont des segments de mémoire virtuelle auxquels sont assignées des portions de fichiers selon une relation de correspondance bit à bit. Ces portions de fichiers reflètent donc exactement les segments correspondant de mémoire virtuelle. Ainsi, en accédant à une zone de mémoire de travail virtuelle, un processus peut accéder à des données du fichier correspondant en utilisant les instructions simples du processeur utilisé qui permettent normalement d'accéder à la mémoire vive, sans exécuter explicitement des opérations d'entrée/sortie. Les opérations d'entrée/sortie nécessaires pour le processus sont automatiquement exécutées par le système d'exploitation, de façon optimisée. Ces opérations sont typiquement mises en œuvre par un mécanisme de pagination auquel font références les expressions couramment utilisées de « *demand paging* » et « *lazy page allocation* ».

Il est observé ici que le mécanisme de gestion de mémoire virtuelle permettant de créer une mémoire virtuelle éparse peut être combiné au mécanisme de gestion de fichiers cartographiés en mémoire pour former un mécanisme de gestion de programmes cartographiés en mémoire (ou *memory mapped programs* en terminologie anglo-saxonne). Ainsi, en combinant ces fonctions, le système d'exploitation mis en œuvre rend disponible le code d'un processus pour l'exécution.

La figure 2 illustre schématiquement un exemple d'un tel mécanisme de chargement de code d'un processus d'une mémoire de masse 120 vers une mémoire de travail virtuelle 200 d'un système informatique tel que celui décrit en référence à la figure 1.

La mémoire de masse 120 comprend ici un ensemble de fichiers dont un fichier 205-1 contenant le code d'un processus à exécuter. Ce fichier peut être découpé en blocs de taille prédéterminée, par exemple en pages.

La mémoire de travail virtuelle 200 utilise une mémoire de travail telle que la mémoire de travail 106 décrite en référence à la figure 1 et une mémoire de masse, par exemple la mémoire de masse 120. La mémoire de travail virtuelle 200 est découpée en régions, une région différente étant dédiée à l'exécution de chaque processus à exécuter. Ainsi, notamment, une région de la mémoire de travail virtuelle 200, référencée 210-1, est affectée à l'exécution du processus correspondant au code mémorisé dans le fichier 205-1.

Lorsque ce processus est appelé pour être exécuté, une première page est chargée du fichier 205-1 en mémoire virtuelle, dans la région 210-1, comme illustré par la référence ①. Lorsqu'au cours de l'exécution du code contenu dans cette page, un appel est effectué vers du code non contenu dans cette page (illustré par la référence ②), c'est-à-dire en présence d'un branchement sur une partie non encore chargée du code, une erreur est générée. En réponse à cette erreur, une instruction de lecture est générée (illustré par la référence ③) et le code correspondant est obtenu pour être exécuté (illustré par la référence ④).

Ainsi, en combinant un mécanisme de gestion de mémoire virtuelle permettant de créer une mémoire virtuelle éparse avec un mécanisme de gestion de fichiers cartographiés en mémoire, il est possible de rendre disponible le code d'un processus pour son exécution.

Par ailleurs, il est rappelé qu'il existe des systèmes de gestion de version de fichiers, généralement appelés *versioning file system* en terminologie anglo-saxonne, pour garder l'historique de modifications effectuées sur des fichiers. Ainsi, après que des modifications aient été validées (par exemple selon une instruction de type *commit*), les modifications sont effectuées dans une nouvelle version du fichier. Il est ainsi possible d'accéder à n'importe quelle version, précédente ou actuelle, du fichier.

Conformément à l'invention, les mécanismes de gestion de mémoire virtuelle permettant de créer une mémoire virtuelle éparse, mémoire éparse,

fichiers cartographiés en mémoire et/ou de gestion de version de fichiers sont combinés pour gérer des données de processus informatiques et ainsi améliorer la gestion d'exécution de processus informatiques.

La figure 3 illustre certaines étapes de gestion d'exécution de processus mises en œuvre conformément à l'invention. Comme illustré, lorsqu'un processus est lancé (étape 300), un espace de mémoire virtuelle éparse est créé (étape 305) ainsi qu'un fichier éparse (étape 310).

L'espace de mémoire virtuelle éparse et le fichier éparse créés sont propres au processus lancé. A chaque fois que le processus est lancé, un espace de mémoire virtuelle éparse et un fichier éparse sont créés. A ces fins, des fonctions standard de gestion d'espaces de mémoires virtuelles éparses et de fichiers éparses sont avantageusement utilisées.

Cet espace de mémoire et ce fichier sont alors liés (étape 315) en utilisant, de préférence, un mécanisme standard de gestion de fichiers cartographiés en mémoire. Cette liaison permet une synchronisation bit à bit entre l'espace de mémoire virtuelle éparse et le fichier éparse correspondant à l'exception des blocs non utilisés.

Le processus est ensuite exécuté de façon standard (étape 320).

Ainsi, lorsqu'un processus est lancé, ses données sont automatiquement placées dans un fichier propre à ce processus. Le contenu de ce fichier évolue automatiquement en fonction de l'évolution des données du processus de telle sorte qu'il y ait une correspondance entre la mémoire virtuelle éparse du processus et le fichier éparse correspondant (une telle synchronisation est représentée par la référence 325 sur la figure 3).

Il est observé ici que l'utilisation de services de contrôle d'accès aux fichiers du système d'exploitation permet de répondre à des exigences de sécurité.

Lors d'une interruption d'exécution du processus considéré (étape 330), volontaire ou non, liée, par exemple, à une fonction de type *checkpoint* ou *suspend* ou à une erreur d'exécution, les données du processus sont présentes dans le fichier éparse précédemment créé et correspondant au processus. En effet, comme décrit précédemment, les données d'un processus évoluant

virtuellement dans le fichier correspondant, il existe une correspondance entre chaque élément des données du processus et les blocs du fichier éparse correspondant. En d'autres termes, dès qu'une donnée est modifiée en mémoire virtuelle (mémoire virtuelle éparse), une modification équivalente est effectuée dans le fichier correspondant (fichier éparse). Ainsi, lors d'une interruption d'exécution, il n'est pas nécessaire de recopier explicitement les données présentes en mémoire virtuelle dans un fichier.

Il est observé ici que les données modifiées en mémoire virtuelle, qui nécessitent une réécriture dans le fichier correspondant, sont dites « *dirty* ». Le système d'exploitation, selon des conditions prédéfinies telles que des conditions de temps passé et de manques de mémoire disponible, « blanchit » la mémoire, c'est-à-dire qu'il réécrit les données dites « *dirty* » dans le fichier correspondant.

Selon un mode de réalisation particulier, un test est alors effectué pour déterminer si un mécanisme de suivi de modifications doit être activé (étape 335), c'est-à-dire s'il convient de garder différentes versions d'un fichier éparse associé à un processus pour conserver l'historique de modification de ces données.

Dans l'affirmative, une instruction de validation (ou *commit*) des données modifiées est générée (étape 340). Cette instruction a pour effet que les prochaines modifications du fichier éparse considéré seront mémorisées dans un nouveau fichier portant le même nom mais ayant une référence de version directement supérieure. A ces fins, une fonction standard de type *versionning* est avantageusement utilisée.

Suite à l'interruption d'exécution d'un processus, il est possible de redémarrer ou reprendre le processus (étape 345), par exemple à l'aide d'une fonction de type *restart* ou *resume*, au point où celui-ci a été interrompu en utilisant le fichier éparse utilisé précédemment pour son exécution.

A ces fins, un nouvel espace de mémoire virtuelle éparse est créé (étape 350) et le dernier fichier éparse utilisé pour l'exécution du processus est obtenu (étape 355). Si un mécanisme de suivi de modifications est activé, un

nouveau fichier éparse est créé pour stocker les modifications apportées aux données.

Les étapes précédentes sont alors répétées (à partir de l'étape 315) : l'espace de mémoire virtuelle éparse créé et le fichier éparse obtenu, ainsi que, le cas échéant, le fichier éparse créé, sont liés pour permettre l'exécution du processus, en utilisant un mécanisme de gestion de fichiers cartographiés en mémoire, comme décrit précédemment.

L'interruption d'un processus peut être volontaire ou involontaire. Une interruption volontaire peut être liée à une raison d'ordre administratif telle que la gestion de priorité d'exécution de plusieurs processus ou technique, notamment pour effectuer une sauvegarde des données et ainsi permettre une analyse des données, par exemple pour des outils de mise au point, et/ou anticiper une éventuelle défaillance. Une interruption involontaire est typiquement liée à une erreur d'interruption. Cependant, quelque soit la nature de l'interruption, le procédé selon l'invention permet une reprise aisée de l'exécution du processus.

En outre, il est observé ici que les données d'un processus évoluant virtuellement dans le fichier éparse qui lui est associé, le mécanisme de gestion de mémoire virtuelle éparse peut y stocker des blocs de données jugés temporairement inutiles et, ultérieurement, en cas de besoin, les faire revenir dans la mémoire de travail (mémoire vive physique). Ainsi, le procédé selon l'invention permet d'éviter le recours à l'usage de la fonction d'échange (*swap*) pour le stockage d'une partie de la mémoire vive virtuelle contrairement aux solutions standard selon lesquels la fonction d'échange est utilisée du fait qu'il n'existe pas d'autre endroit où stocker une partie de la mémoire de travail virtuelle.

Naturellement, pour satisfaire des besoins spécifiques, une personne compétente dans le domaine de l'invention pourra appliquer des modifications dans la description précédente.

REVENDICATIONS

1. Procédé de gestion d'exécution d'au moins un processus dans un ordinateur comprenant une mémoire de travail et une mémoire de masse, ledit ordinateur comprenant un système d'exploitation configuré pour gérer de la mémoire de travail virtuelle, ce procédé étant caractérisé en ce qu'il comprend les étapes suivantes lors du lancement dudit au moins un processus,

- création (305) d'au moins une région de mémoire virtuelle éparse propre audit au moins un processus lancé ;
- création (310) d'au moins un fichier éparse associé audit au moins un processus lancé ; et,
- établissement (315) d'un lien de correspondance entre ladite au moins une région de mémoire virtuelle éparse et ledit au moins un fichier éparse créés, selon une fonction de gestion de fichiers cartographiés en mémoire.

2. Procédé selon la revendication 1 comprenant en outre une étape d'exécution (320) dudit au moins un processus, selon lequel, durant ladite étape d'exécution dudit au moins un processus, ledit système d'exploitation copie (325) des données associées audit au moins un processus et mémorisées dans ladite au moins une région de mémoire virtuelle éparse créée vers ledit au moins un fichier éparse créé.

3. Procédé selon la revendication 2 comprenant en outre, suite à la validation d'une modification d'une donnée associée audit au moins un processus et mémorisée dans ladite au moins une région de mémoire virtuelle éparse créée et recopiée dans ledit au moins un fichier éparse créé, une étape de création d'une nouvelle version dudit au moins un fichier éparse créé.

4. Procédé selon l'une quelconque des revendications 1 à 3 comprenant en outre une étape d'interruption (330) d'exécution dudit au moins un processus.

5. Procédé selon la revendication 4 comprenant en outre, suite à ladite étape d'interruption d'exécution, une étape de reprise (345) d'exécution dudit au moins un processus.

6. Procédé selon la revendication 5 selon lequel ladite étape de reprise d'exécution dudit au moins un processus comprend les étapes suivantes,

- création (350) d'au moins une région de mémoire virtuelle éparse propre audit au moins un processus relancé ;
- obtention (355) dudit au moins un fichier éparse préalablement créé et associé audit au moins un processus lancé ; et,
- établissement (315) d'un lien de correspondance entre ladite au moins une région de mémoire virtuelle éparse créé propre audit au moins un processus relancé et ledit au moins un fichier éparse obtenu, selon une fonction de gestion de fichiers cartographiés en mémoire.

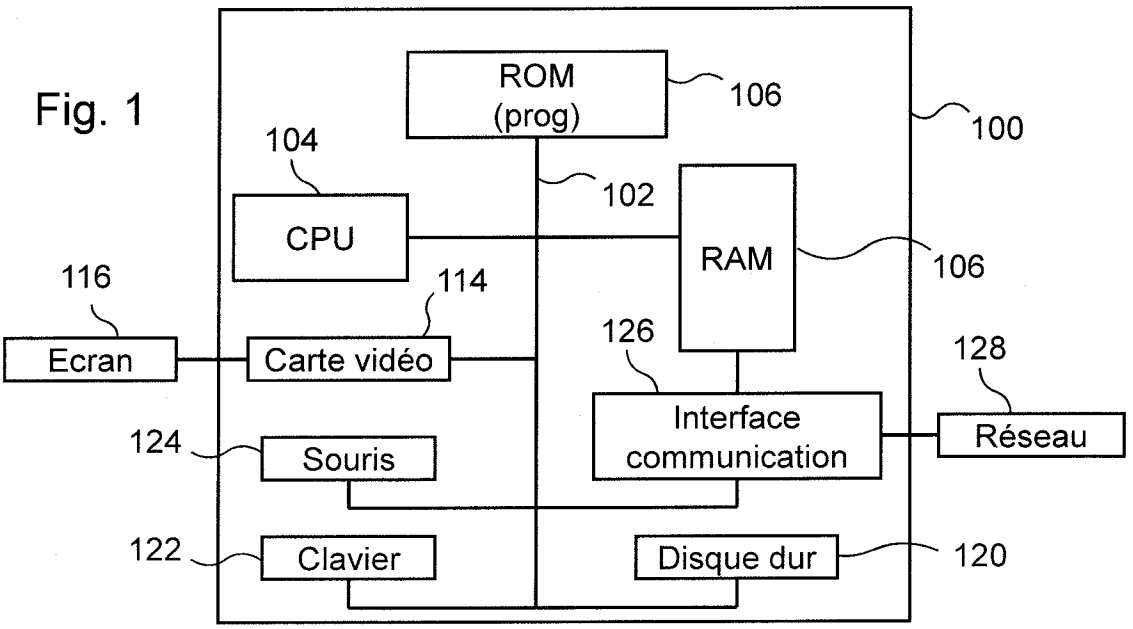
7. Procédé selon l'une quelconque des revendications 1 à 6 selon lequel au moins l'une des étapes de création d'au moins une région de mémoire virtuelle éparse, de création d'au moins un fichier éparse et d'établissement d'un lien de correspondance entre une région de mémoire virtuelle éparse et un fichier éparse est une fonction dudit système d'exploitation.

8. Procédé selon l'une quelconque des revendications 1 à 7 utilisé dans un outil de mise au point d'au moins un processus.

9. Programme d'ordinateur comprenant des instructions adaptées à la mise en œuvre de chacune des étapes du procédé selon l'une quelconque des revendications précédentes lorsque ledit programme est exécuté sur un ordinateur.

10. Dispositif comprenant des moyens adaptés à la mise en œuvre de chacune des étapes du procédé selon l'une quelconque des revendications 1 à 8.

Fig. 1


ROM
(prog)
106
100
104
102
CPU
RAM
106
116
114
126
128
Ecran
Carte vidéo
Interface
communication
Réseau
124
Souris
122
Clavier
Disque dur
120

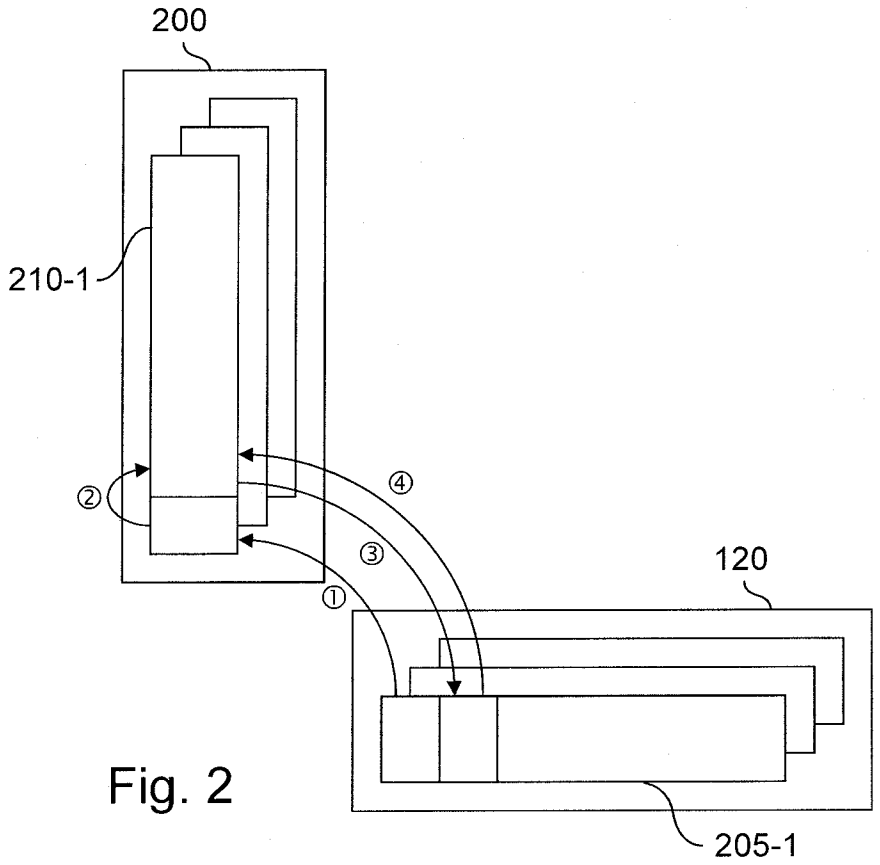

200
210-1
②
④
③
①
120
205-1

Fig. 2

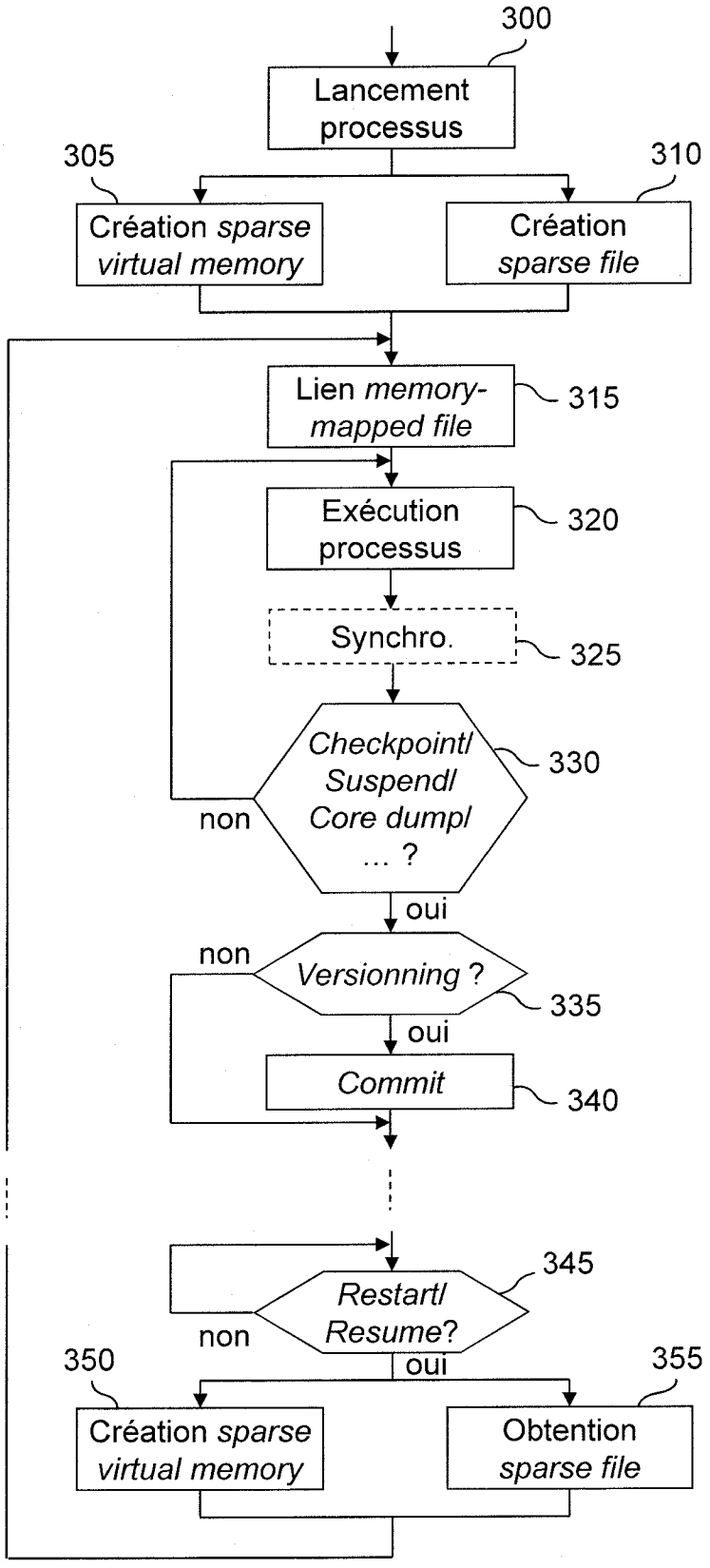

300
Lancement processus
305
310
Création sparse virtual memory
Création sparse file
Lien memory-mapped file
315
Exécution processus
320
Synchro.
325
Checkpoint/ Suspend/ Core dump/ ... ?
330
non
oui
non
Versionning ?
335
oui
Commit
340
345
Restart/ Resume?
non
oui
350
355
Création sparse virtual memory
Obtention sparse file

Fig. 3

INTERNATIONAL SEARCH REPORT

International application No
PCT/FR2013/050297

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F12/10 G06F11/36
ADD.

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

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F

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

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2004/024729 A1 (WORLEY JOHN S [US]) 5 February 2004 (2004-02-05)	1,7-10
A	paragraphs [0002], [0004], [0016], [0017], [0018], [0021], [0022] - [0033]; claim 1; figures 1-6	2-6

A	DULCARDO ARTEAGA ET AL: "Towards Scalable Application Checkpointing with Parallel File System Delegation", NETWORKING, ARCHITECTURE AND STORAGE (NAS), 2011 6TH IEEE INTERNATIONAL CONFERENCE ON, IEEE, 28 July 2011 (2011-07-28), pages 130-139, XP032048443, DOI: 10.1109/NAS.2011.42 ISBN: 978-1-4577-1172-5 the whole document	1-10

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents :

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

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

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

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

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

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

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

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

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
28 March 2013	08/04/2013

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Jardon, Stéphan

5

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/FR2013/050297

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2004024729 A1	05-02-2004	NONE	

RAPPORT DE RECHERCHE INTERNATIONALE

Demande internationale n°
PCT/FR2013/050297

A. CLASSEMENT DE L'OBJET DE LA DEMANDE
INV. G06F12/10 G06F11/36
ADD.

Selon la classification internationale des brevets (CIB) ou à la fois selon la classification nationale et la CIB

B. DOMAINES SUR LESQUELS LA RECHERCHE A PORTE

Documentation minimale consultée (système de classification suivi des symboles de classement)
G06F

Documentation consultée autre que la documentation minimale dans la mesure où ces documents relèvent des domaines sur lesquels a porté la recherche

Base de données électronique consultée au cours de la recherche internationale (nom de la base de données, et si cela est réalisable, termes de recherche utilisés)
EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERES COMME PERTINENTS

Catégorie*	Identification des documents cités, avec, le cas échéant, l'indication des passages pertinents	no. des revendications visées
X	US 2004/024729 A1 (WORLEY JOHN S [US]) 5 février 2004 (2004-02-05)	1,7-10
A	alinéas [0002], [0004], [0016], [0017], [0018], [0021], [0022] - [0033]; revendication 1; figures 1-6	2-6

A	DULCARDO ARTEAGA ET AL: "Towards Scalable Application Checkpointing with Parallel File System Delegation", NETWORKING, ARCHITECTURE AND STORAGE (NAS), 2011 6TH IEEE INTERNATIONAL CONFERENCE ON, IEEE, 28 juillet 2011 (2011-07-28), pages 130-139, XP032048443, DOI: 10.1109/NAS.2011.42 ISBN: 978-1-4577-1172-5 le document en entier	1-10

☐ Voir la suite du cadre C pour la fin de la liste des documents

☒ Les documents de familles de brevets sont indiqués en annexe

* Catégories spéciales de documents cités:

"A" document définissant l'état général de la technique, non considéré comme particulièrement pertinent

"E" document antérieur, mais publié à la date de dépôt international ou après cette date

"L" document pouvant jeter un doute sur une revendication de priorité ou cité pour déterminer la date de publication d'une autre citation ou pour une raison spéciale (telle qu'indiquée)

"O" document se référant à une divulgation orale, à un usage, à une exposition ou tous autres moyens

"P" document publié avant la date de dépôt international, mais postérieurement à la date de priorité revendiquée

"T" document ultérieur publié après la date de dépôt international ou la date de priorité et n'appartenant pas à l'état de la technique pertinent, mais cité pour comprendre le principe ou la théorie constituant la base de l'invention

"X" document particulièrement pertinent; l'invention revendiquée ne peut être considérée comme nouvelle ou comme impliquant une activité inventive par rapport au document considéré isolément

"Y" document particulièrement pertinent; l'invention revendiquée ne peut être considérée comme impliquant une activité inventive lorsque le document est associé à un ou plusieurs autres documents de même nature, cette combinaison étant évidente pour une personne du métier

"&" document qui fait partie de la même famille de brevets

Date à laquelle la recherche internationale a été effectivement achevée	Date d'expédition du présent rapport de recherche internationale
28 mars 2013	08/04/2013

Nom et adresse postale de l'administration chargée de la recherche internationale	Fonctionnaire autorisé
Office Européen des Brevets, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Jardon, Stéphan

5

RAPPORT DE RECHERCHE INTERNATIONALE

Renseignements relatifs aux membres de familles de brevets

Demande internationale n°
PCT/FR2013/050297

Document brevet cité au rapport de recherche	Date de publication	Membre(s) de la famille de brevet(s)	Date de publication
US 2004024729 A1	05-02-2004	AUCUN	