(19) RÉPUBLIQUE FRANÇAISE

INSTITUT NATIONAL
DE LA PROPRIÉTÉ INDUSTRIELLE

PARIS

(11) **N° de publication :** **2 871 255**
(à n'utiliser que pour les commandes de reproduction)

(21) **N° d'enregistrement national :** **04 06076**

(51) Int Cl[7] : **G 06 F 9/45**

# (12) DEMANDE DE BREVET D'INVENTION A1

(22) **Date de dépôt :** 04.06.04.

(30) **Priorité :**

(43) **Date de mise à la disposition du public de la demande :** 09.12.05 Bulletin 05/49.

(56) **Liste des documents cités dans le rapport de recherche préliminaire :** *Se reporter à la fin du présent fascicule*

(60) **Références à d'autres documents nationaux apparentés :**

(71) **Demandeur(s) :** *GEMPLUS Société anonyme* — FR.

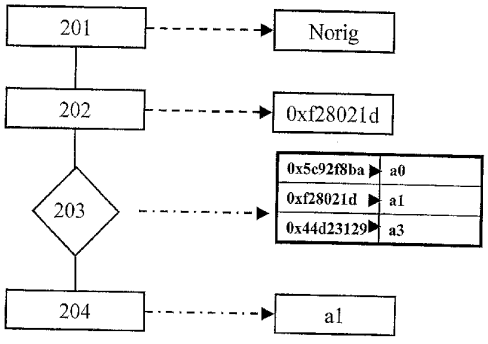
(72) **Inventeur(s) :** PEIRANI BEATRICE et VICTOR LIONEL.

(73) **Titulaire(s) :**

(74) **Mandataire(s) :** NOVAGRAAF TECHNOLOGIES (CABINET BALLOT).

## (54) PROCEDE D'OBFUSCATION DE CODE COMPILE, TERMINAL ET LOGICIEL ASSOCIES.

(57) L'invention propose un procédé d'obfuscation d'un logiciel compilé comprenant des étapes de:
-remplacement (103) d'au moins un nom d'un composant logiciel par un nom brouillé;
-insertion d'un module (105) prévu pour, lors de l'exécution du logiciel:
-intercepter un appel à une fonction de réflexion appliquée au nom du composant logiciel;
-générer, par un calcul irréversible et prédéterminé, un identifiant du nom du composant;
-retrouver le nom brouillé au moyen de l'identifiant; et
-commander l'exécution de la fonction de réflexion appliquée au nom brouillé.




FR 2 871 255 - A1

# PROCEDE D'OBFUSCATION DE CODE COMPILE, TERMINAL ET LOGICIEL ASSOCIES

La présente invention porte sur les logiciels compilés dans un langage intermédiaire de programmation et plus particulièrement sur les procédés d'obfuscation de tels logiciels.

Diverses motivations poussent des personnes à décompiler un logiciel pour obtenir son code source : comprendre le fonctionnement du logiciel, intervenir frauduleusement sur un algorithme de cryptage ou encore récupérer des routines complètes pour les intégrer dans ses propres logiciels.

L'intervention sur des algorithmes de cryptage peut viser à décrypter frauduleusement des données. La récupération de routines complètes peut avoir pour but de réduire les efforts de développement de logiciels concurrents : il peut alors suffire d'intégrer ces routines dans son propre logiciel pour réaliser des fonctions essentielles du logiciel. L'intégration partielle d'un logiciel est particulièrement difficile à prouver pour l'auteur du logiciel initial. Ainsi, l'auteur a généralement de très grandes difficultés à faire valoir ses droits auprès d'un tribunal.

Des techniques d'obfuscation ont ainsi été développées. L'obfuscation définit tout procédé appliqué au code compilé du logiciel pour le rendre moins facilement intelligible lors de la décompilation, sans influer sur son fonctionnement.

Des langages intermédiaires de programmation ont été développés. Le but principal des logiciels dans ces

langages est de les rendre indépendants du matériel sur lequel ils doivent être exécutés. Lors de l'exécution, de tels logiciels doivent utiliser des composants logiciels dont le nom est aisément intelligible ou correspond à un nom public. Ainsi, de nombreux éléments du code source peuvent être identifiés avec une très grande facilité lors de la décompilation.

Un certain nombre d'outils d'obfuscation ont déjà été mis sur le marché pour rendre des logiciels en Java moins intelligibles, tout en respectant une exécution à l'identique. La plupart de ces outils utilisent la technique d'obfuscation de la disposition (Layout obfuscation en langue anglaise). Cette technique consiste par exemple à modifier le logiciel compilé pour que les informations non nécessaires à son exécution soient masquées et/ou supprimées dans son code compilé : on retire par exemple les informations de débuggage, on retire les commentaires des programmateurs ou on altère le nom de composants logiciels.

Le nom utilisé lors de l'exécution pour certains composants logiciels (par exemple des fonctions, des méthodes ou des variables) n'est pas toujours prévisible au moment de la compilation du logiciel en langage intermédiaire. Un tel nom peut en effet être généré dynamiquement au moment de l'exécution du logiciel par une machine virtuelle. Par une ou plusieurs fonctions de réflexion, on génère, à partir du nom du composant, un nom de composant logiciel utilisable au moment de l'exécution. Afin de faciliter la mise en œuvre de ces fonctions de réflexion , le nom

généré est très proche du nom du composant logiciel. La génération de ce nom peut notamment être obtenue par permutation des caractères du nom d'un composant logiciel. Ce type de transformation est donc aisément réversible. Le niveau d'obfuscation ainsi présenté est donc réduit.

L'invention vise à résoudre cet inconvénient. L'invention a ainsi pour objet un procédé d'obfuscation d'un logiciel compilé dans un langage intermédiaire présentant des fonctions de réflexion, ce procédé comprenant des étapes de :

-remplacement d'au moins un nom d'un composant logiciel dans le code compilé par un nom brouillé ;

-insertion d'un module dans le code compilé, ce module étant prévu pour, lors de l'exécution du logiciel :

-intercepter un appel à une fonction de réflexion appliquée au nom du composant logiciel;

-générer, par un calcul irréversible et prédéterminé, un identifiant du nom du composant;

-retrouver le nom brouillé au moyen de l'identifiant ; et

-commander l'exécution de la fonction de réflexion appliquée au nom brouillé.

Selon une variante, ce procédé comprend les étapes de :

-génération de l'identifiant du nom du composant par le calcul prédéterminé ;

-mémorisation d'un lien entre l'identifiant et le nom brouillé dans le code compilé ;

-le module inséré est prévu pour, lors de l'exécution :

-recevoir un nom pour lequel une fonction de réflexion doit être appliquée ;

-générer par ledit calcul un identifiant du nom reçu;

-comparer cet identifiant à l'identifiant mémorisé ;

-si les identifiants correspondent, récupérer le nom brouillé au moyen du lien et commander l'exécution de la fonction de réflexion appliquée au nom brouillé ; et

-sinon, commander l'exécution de la fonction de réflexion appliquée au nom reçu.

Selon une autre variante, les étapes de génération de l'identifiant et de mémorisation du lien sont réalisées pour plusieurs noms de composants logiciels, et les liens sont mémorisés dans au moins une table associant un identifiant généré au nom brouillé correspondant.

Selon encore une variante, plusieurs tables de mémorisation sont générées et un identifiant généré et un nom brouillé correspondant sont stockés dans une table choisie en fonction du nom du composant associé.

Selon un mode de réalisation, le calcul prédéterminé pour la génération de l'identifiant du nom du composant comprend le hachage de ce nom.

Selon une variante, le hachage d'un nom comprend :

-un premier hachage du nom du composant à partir duquel on détermine la table où l'on stocke l'identifiant associé à ce nom ;

-un second hachage du nom du composant, distinct du premier, dont le résultat constitue l'identifiant associé à ce nom dans la table déterminée.

Selon un autre mode de réalisation, le procédé comprend en outre une étape d'inscription d'une clé de cryptage dans une mémoire à accès restreint, le calcul prédéterminé est un cryptage mettant en œuvre ladite clé.

L'invention porte également sur un logiciel présentant des composants logiciels, compilé dans un langage intermédiaire présentant des fonctions de réflexion, ce logiciel étant obfusqué par un tel procédé.

L'invention porte encore sur un objet électronique qui comprend des moyens de mise en œuvre d'un tel logiciel. on peut également prévoir un objet électronique comprenant un support stockant un tel logiciel. Cet objet peut être un terminal, un combiné de téléphonie mobile ou une carte à puce.

D'autres particularités et avantages de l'invention apparaîtront clairement à la lecture de la description faite à titre d'exemple non limitatif et en regard des dessins annexés sur lesquels :

-la figure 1 illustre des étapes d'un mode de réalisation d'un procédé d'obfuscation selon l'invention ;

-les figures 2 et 3 illustrent le fonctionnement d'un logiciel obfusqué selon un mode de réalisation de l'invention.

Des exemples de composants logiciels sont les fonctions, les noms de variable ou les méthodes.

En programmation orientée objet, une fonction de réflexion désigne généralement un mécanisme permettant à un conteneur de lire les caractéristiques d'un composant logiciel à l'aide de conventions de nom. La réflexion permet de lire les propriétés, les méthodes voire les évènements d'un composant.

Une fonction de hachage « H » transforme une entrée de données d'une dimension variable « m » et donne comme résultat une sortie de données de taille fixe « h » (h=H(m)). Ainsi, une fonction de hachage doit remplir quelques conditions de base :

-l'entrée peut être de dimension variable ;

-la sortie doit être de dimension fixe ;

-H(m) doit être relativement facile à calculer.

Un exemple d'une telle fonction de hachage est la fonction qui, à une suite d'octets, fait correspondre le XOR de tous les octets.

Une fonction de hachage à usage cryptographique doit respecter les conditions suivantes :

-H(m) doit être une fonction à sens unique (la sortie de la fonction est aisément calculable à partir de l'entrée, mais il est extrêmement difficile de trouver une entrée qui se hache en une sortie donnée) ;

-H(m) doit être « sans collision » : (il est difficile de générer deux entrées ayant la même valeur de haché en sortie de la fonction).

Une fonction de hachage à usage cryptographique est par exemple la fonction SHA-1.

L'invention propose d'obfusquer un logiciel compilé dans un langage intermédiaire présentant des fonctions de réflexion en améliorant la robustesse d'obfuscation des noms de composants logiciels. Pour cela, le procédé comprend une étape de remplacement du nom d'un composant logiciel par un nom brouillé. Un module fonctionnel est alors inséré dans le code compilé. Ce module fonctionnel réalise les opérations suivantes lors de l'exécution du logiciel :

-interception d'un appel à une fonction de réflexion pour un nom de composant logiciel ;

-génération d'un identifiant de ce nom par un calcul irréversible et prédéterminé effectué sur ce nom;

-récupération du nom brouillé au moyen de l'identifiant ;

-commande de l'exécution de la fonction de réflexion appliquée au nom brouillé.

Ce procédé d'obfuscation fournit un code compilé avec des noms brouillés de façon irréversible, et dont l'exécution n'a pas d'incidence sur les fonctions de réflexion. Le module permet en effet de retrouver le nom brouillé lors de l'exécution mais il n'existe pas de moyen de retrouver le nom du composant à partir de ce nom brouillé.

On peut prévoir que l'invention s'applique à des langages intermédiaires présentant des fonctions de réflexion, tels que Java ou .Net (marques déposées).

Le diagramme de la figure 1 illustre un exemple d'un tel procédé. Lors d'une étape 101, le procédé d'obfuscation est lancé pour être appliqué à un logiciel compilé. Lors d'une étape 102, on sélectionne un nom d'un composant logiciel dans le code compilé. Lors d'une étape 103, le nom est remplacé par un nom brouillé. Les étapes 102 et 103 peuvent être exécutées pour différents noms de composants du logiciel compilé. Facultativement, à l'étape 104, on génère par un calcul irréversible des identifiants des noms et on mémorise dans le code compilé un lien entre les identifiants et les noms brouillés. Lors de l'étape 105, le module fonctionnel décrit auparavant est inséré dans le code compilé.

Le module inséré présente alors les fonctions suivantes : lors de chaque appel à une fonction de réflexion, le calcul prédéterminé est réalisé sur le nom pour lequel la fonction de réflexion doit être appliquée. Cette étape correspond au calcul qui a été réalisé pour générer des identifiants lors de l'obfuscation. Le module compare l'identifiant qui vient d'être calculé aux identifiants qui ont été mémorisés lors de l'obfuscation. Si cet identifiant a été mémorisé au préalable, cela signifie que le nom correspondant a été remplacé par un nom brouillé dans le code compilé. La fonction de réflexion est alors appliquée au nom brouillé. Par contre, si aucun identifiant mémorisé ne correspond à l'identifiant qui vient d'être calculé, on déduit que le nom n'a pas été modifié lors de l'obfuscation. La fonction de réflexion est alors appliquée au nom.

Ainsi, la réflexion s'applique aux noms de composants modifiés et aux noms de composants non modifiés. Les noms non modifiés par l'obfuscation sont typiquement des noms publics, mis à la disposition de composants logiciels extérieurs au logiciel compilé, ou encore des noms générés dynamiquement lors de l'exécution et qui ne sont pas connus lors de la compilation.

On peut prévoir que les liens soient mémorisés dans le code compilé sous forme d'une ou plusieurs tables de correspondance associant un nom brouillé à un identifiant calculé lors de l'obfuscation. On prévoit que la ou les tables ne soient accessibles que par le module d'obfuscation inséré.

Selon une variante préférentielle, le calcul prédéterminé est un hachage. Le nom d'origine est ainsi impossible à retrouver à partir du nom haché. Un exemple détaillé du fonctionnement d'un module du code compilé pour cette variante est illustré aux figures 2 et 3. On a illustré à gauche de ces figures les opérations qui était réalisées et à droite les éléments principaux utilisés lors de ces opérations.

La figure 2 illustre le cas où le nom (pour lequel la fonction de réflexion doit être appliquée) a été modifié lors de l'obfuscation. A l'étape 201, l'appel à la fonction de réflexion est intercepté et le module récupère le nom Norig pour lequel la réflexion devait être appliquée. A l'étape 202, un hachage du nom Norig est calculé. On obtient un identifiant 0xf28021d. A l'étape 203 on vérifie la présence de cet identifiant dans une table de correspondance entre les identifiants

et les noms brouillés correspondants. Dans l'exemple, le module détermine que l'identifiant 0fx28021d est présent dans la table et récupère donc le nom brouillé associé : a1. A l'étape 204, la réflexion est appliquée au nom brouillé a1.

La figure 3 illustre le cas où le nom pour lequel la réflexion doit être appliquée n'a pas été modifié lors de l'obfuscation. A l'étape 301, l'appel à la fonction de réflexion est intercepté et le module récupère le nom Ndyna pour lequel la réflexion devait être appliquée. A l'étape 302, un hachage du nom Ndyna est calculé. On obtient un identifiant 0x020400b. A l'étape 303 on détermine que cet identifiant n'est pas présent dans la table de correspondance. A l'étape 304, la réflexion est donc appliquée au nom Ndyna.

Si un utilisateur examine les noms des composants logiciels qui passent lors de l'exécution, il pourra au mieux obtenir la valeur de hachage correspondante dans la table. Cependant, cette valeur ne lui permet pas de déduire le nom du composant pour lequel la fonction de réflexion a été appelée.

Il faut également noter qu'en programmation orientée objet, deux champs appartenant à des classes différentes peuvent porter le même nom avant l'obfuscation. Pour éviter que ces champs n'aboutissent à un même identifiant après obfuscation, le calcul prédéterminé doit être appliqué au nom complet d'un attribut. Pour un champ, on hachera par exemple une chaîne de caractères composée du nom de la classe et du nom du champ. Pour une méthode, on hachera une chaîne

de caractères composée du nom de la méthode et d'un complément représentatif des arguments de la méthode.

Dans une application particulière au langage Java, on peut prévoir d'utiliser la classe Java String pour réaliser la fonction de hachage. Ainsi, pour un nom de composant s à hacher, la fonction de hachage aboutira à une valeur de 32 bit obtenue de la façon suivante :

S[0]*31^(n-1)+s[2]*31^(n-2)…+s[n-1]

La classe Java Hashtable peut être utilisée pour former une table de correspondance.

Le module inséré dans le code compilé peut être réalisé sous forme de classe propriétaire java.lang.class. Cette classe peut contenir une copie de la table de correspondance. Lors d'appels à des méthodes de java.lang.class impliquant une fonction de réflexion, la classe propriétaire intercepte cette réflexion. Cette classe propriétaire effectue les hachages et effectue les tests de présence des identifiants dans les tables de correspondance.

Pour limiter encore les informations qui pourraient être déduites du code compilé, on utilise plusieurs tables de correspondance. L'identifiant et le nom brouillé sont stockés dans une table choisie en fonction du nom brouillé, du nom du composant ou de l'identifiant.

Dans cette variante, on peut prévoir que le calcul comprenne plusieurs fonctions de hachage. On peut prévoir qu'une première fonction de hachage sera appliquée à un nom, pour déterminer dans quelle table

son identifiant devrait être stocké. Une seconde fonction de hachage serait alors appliquée à ce nom pour obtenir l'identifiant associé à ce nom. Cette variante permet d'obtenir un gain en sécurité qui compense largement le temps de traitement supplémentaire généré.

Selon un autre mode de réalisation, l'obfuscation implique l'insertion d'une clé préférentiellement dans une mémoire à accès restreint. Il peut par exemple s'agir d'une mémoire amovible et externe à un appareil électronique mémorisant le logiciel obfusqué. Cette mémoire peut être incluse dans une clé USB, une carte à puce ou tout autre support mémoire similaire. Cette mémoire peut également être incluse dans l'appareil électronique. Selon une variante, la clé est insérée dans le code compilé.

Le calcul prédéterminé met alors en œuvre un procédé de cryptage à l'aide de cette clé. On calcule des noms brouillés par cryptage de plusieurs noms de composants puis on mémorise une liste de ces noms qui ont fait l'objet d'un remplacement lors de l'obfuscation. Si le nom pour lequel une réflexion doit être appliquée est présent dans la liste, la réflexion est appliquée au nom brouillé ainsi obtenu. Si ce nom n'est pas présent dans la liste, la réflexion lui est appliquée.

Un logiciel compilé ainsi obfusqué peut par exemple être mis en œuvre dans un objet électronique adapté. Un tel objet électronique peut présenter un support de stockage mémorisant ce logiciel. Ce logiciel peut

notamment être stocké sous forme d'applet Java. L'objet électronique peut notamment être un terminal, un combiné de téléphonie mobile ou une carte à puce.

## REVENDICATIONS

1. Procédé d'obfuscation d'un logiciel compilé dans un langage intermédiaire présentant des fonctions de réflexion, caractérisé en ce que le procédé comprend des étapes de :

-remplacement (103) d'au moins un nom d'un composant logiciel dans le code compilé par un nom brouillé ;

-insertion d'un module (105) dans le code compilé, ce module étant prévu pour, lors de l'exécution du logiciel :

-intercepter un appel à une fonction de réflexion appliquée au nom du composant logiciel;

-générer, par un calcul irréversible et prédéterminé, un identifiant du nom du composant;

-retrouver le nom brouillé au moyen de l'identifiant ; et

-commander l'exécution de la fonction de réflexion appliquée au nom brouillé.

2. Procédé d'obfuscation selon la revendication 1, caractérisé en ce que :

-ce procédé comprend les étapes de :

-génération (104) de l'identifiant du nom du composant par le calcul prédéterminé ;

-mémorisation d'un lien entre l'identifiant et le nom brouillé dans le code compilé ;

-le module inséré est prévu pour, lors de l'exécution :

-recevoir un nom (Norig) pour lequel une fonction de réflexion doit être appliquée ;

-générer par ledit calcul un identifiant du nom reçu;

-comparer cet identifiant à l'identifiant mémorisé ;

-si les identifiants correspondent, récupérer le nom brouillé (a1) au moyen du lien et commander l'exécution de la fonction de réflexion appliquée au nom brouillé ; et

-sinon, commander l'exécution de la fonction de réflexion appliquée au nom reçu.

3. Procédé d'obfuscation selon la revendication 2, caractérisé en ce que les étapes de génération de l'identifiant et de mémorisation du lien sont réalisées pour plusieurs noms de composants logiciels, et en ce que les liens sont mémorisés dans au moins une table associant un identifiant généré au nom brouillé correspondant.

4. Procédé d'obfuscation selon la revendication 3, caractérisé en ce que plusieurs tables de mémorisation sont générées et en ce qu'un identifiant généré et un nom brouillé correspondant sont stockés dans une table choisie en fonction du nom du composant associé.

5. Procédé d'obfuscation selon l'une quelconque des revendications précédentes, caractérisé en ce que le calcul prédéterminé pour la génération de

l'identifiant du nom du composant comprend le hachage de ce nom.

6. Procédé d'obfuscation selon les revendications 4 et 5, caractérisé en ce que le hachage d'un nom comprend :

-un premier hachage du nom du composant à partir duquel on détermine la table où l'on stocke l'identifiant associé à ce nom ;

-un second hachage du nom du composant, distinct du premier, dont le résultat constitue l'identifiant associé à ce nom dans la table déterminée.

7. Procédé d'obfuscation selon l'une quelconque des revendications 1 à 4, caractérisé en ce que :

-le procédé comprend en outre une étape d'inscription d'une clé de cryptage dans une mémoire à accès restreint ;

-le calcul prédéterminé est un cryptage mettant en œuvre ladite clé.

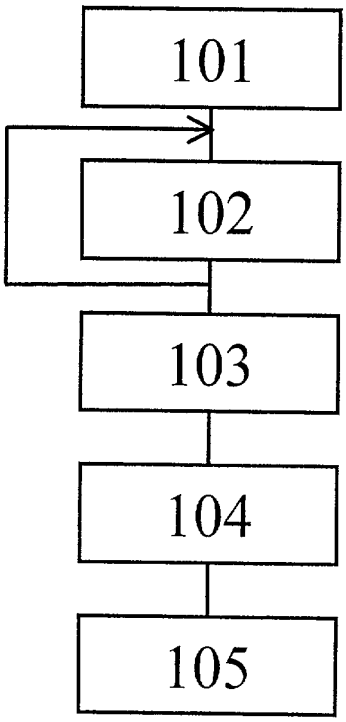



Fig. 1

Fig. 2

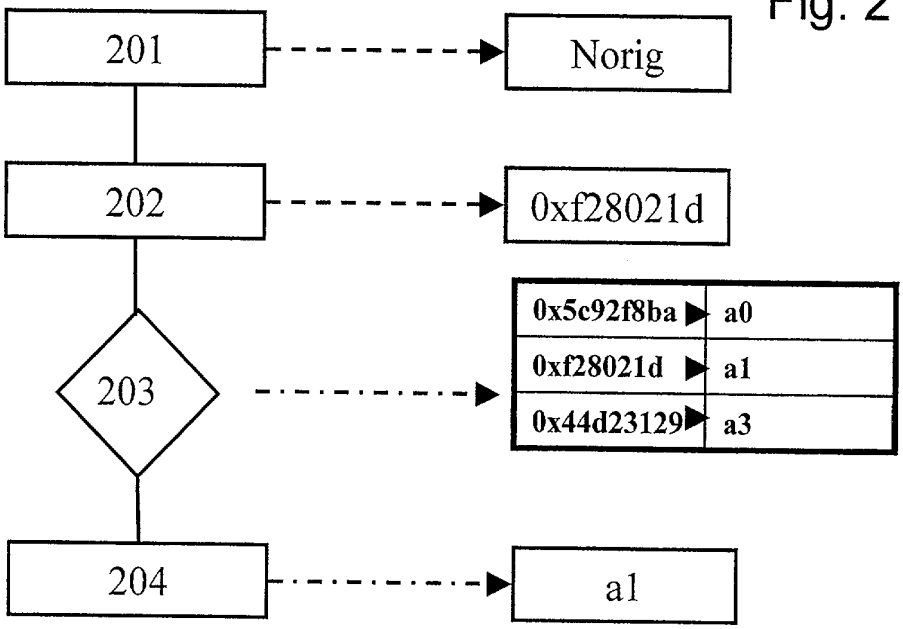
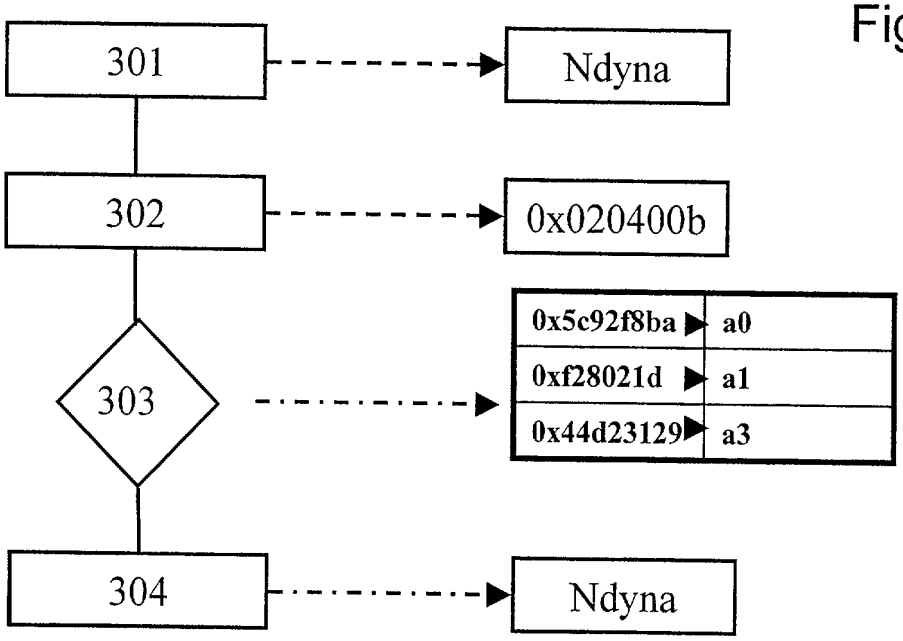

201
Norig
202
0xf28021d
203
0x5c92f8ba
a0
0xf28021d
a1
0x44d23129
a3
204
a1

Fig. 3


301
Ndyna
302
0x020400b
303
0x5c92f8ba
a0
0xf28021d
a1
0x44d23129
a3
304
Ndyna

RÉPUBLIQUE FRANÇAISE

2871255

INPI
INSTITUT NATIONAL DE LA PROPRIETE INDUSTRIELLE

# RAPPORT DE RECHERCHE PRÉLIMINAIRE

établi sur la base des dernières revendications déposées avant le commencement de la recherche

N° d'enregistrement national

FA 654379
FR 0406076

| DOCUMENTS CONSIDÉRÉS COMME PERTINENTS | | Revendication(s) concernée(s) | Classement attribué à l'invention par l'INPI |
|---|---|---|---|
| Catégorie | Citation du document avec indication, en cas de besoin, des parties pertinentes | | |
| A | WO 00/72112 A2 (FRAUNHOFER CRCG, INC; LUO, CHENGHUI; ZHAO, JIAN)<br>30 novembre 2000 (2000-11-30)<br>* abrégé *<br>* page 3, ligne 31 - page 5, ligne 9 *<br>* page 7, ligne 23 - page 10, ligne 15 *<br>* page 11, ligne 24 - page 12, ligne 9 *<br>* figures 2-7 *<br>----- | 1-7 | G06F9/45 |
| A | WO 2004/023313 A1 (FRAUNHOFER CRCG, INC; LUO, CHENGHUI; ZHAO, JIAN)<br>18 mars 2004 (2004-03-18)<br>* abrégé *<br>* page 3, ligne 26 - page 5, ligne 9 *<br>* page 7, ligne 1 - page 8, ligne 30 *<br>* page 12, ligne 6 - page 14, ligne 22 *<br>* figures 2-6,14-16 *<br>----- | 1-7 | |
| A | US 6 668 325 B1 (COLLBERG CHRISTIAN SVEN ET AL) 23 décembre 2003 (2003-12-23)<br>* abrégé *<br>* colonne 1, ligne 64 - colonne 3, ligne 20 *<br>* colonne 4, ligne 36 - colonne 7, ligne 34 *<br>* colonne 15, ligne 18 - colonne 16, ligne 35 *<br>* figures 2D,2E,5,6 *<br>----- | 1-7 | DOMAINES TECHNIQUES RECHERCHÉS (Int.CL.7)<br>G06F |

| Date d'achèvement de la recherche | Examinateur |
|---|---|
| 26 janvier 2005 | Bichler, M |

CATÉGORIE DES DOCUMENTS CITÉS

X : particulièrement pertinent à lui seul
Y : particulièrement pertinent en combinaison avec un autre document de la même catégorie
A : arrière-plan technologique
O : divulgation non-écrite
P : document intercalaire

T : théorie ou principe à la base de l'invention
E : document de brevet bénéficiant d'une date antérieure à la date de dépôt et qui n'a été publié qu'à cette date de dépôt ou qu'à une date postérieure.
D : cité dans la demande
L : cité pour d'autres raisons

& : membre de la même famille, document correspondant



EPO FORM 1503 12.99 (P04C14)

## ANNEXE AU RAPPORT DE RECHERCHE PRÉLIMINAIRE RELATIF A LA DEMANDE DE BREVET FRANÇAIS NO. FR 0406076 FA 654379

La présente annexe indique les membres de la famille de brevets relatifs aux documents brevets cités dans le rapport de recherche préliminaire visé ci-dessus.
Les dits membres sont contenus au fichier informatique de l'Office européen des brevets à la date du 26-01-2005
Les renseignements fournis sont donnés à titre indicatif et n'engagent pas la responsabilité de l'Office européen des brevets, ni de l'Administration française

| Document brevet cité au rapport de recherche | Date de publication | Membre(s) de la famille de brevet(s) | Date de publication |
|---|---|---|---|
| WO 0072112 A2 | 30-11-2000 | US 2004172544 A1 | 02-09-2004 |
| WO 2004023313 A1 | 18-03-2004 | US 2004172544 A1 | 02-09-2004 |
| US 6668325 B1 | 23-12-2003 | AU 7957998 A | 25-01-1999 |
| | | CA 2293650 A1 | 14-01-1999 |
| | | CN 1260055 T | 12-07-2000 |
| | | EP 0988591 A1 | 29-03-2000 |
| | | JP 2002514333 T | 14-05-2002 |
| | | WO 9901815 A1 | 14-01-1999 |

EPO FORM P0465