

①②

DEMANDE DE BREVET D'INVENTION

A1

②② Date de dépôt : 27.11.97.

③⑦ Priorité :

④③ Date de mise à la disposition du public de la demande : 28.05.99 Bulletin 99/21.

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

⑥⑦ Références à d'autres documents nationaux apparentés :

⑦① Demandeur(s) : I-CARD Société anonyme — FR.

⑦② Inventeur(s) : BARTH CHRISTOPHE.

⑦③ Titulaire(s) :

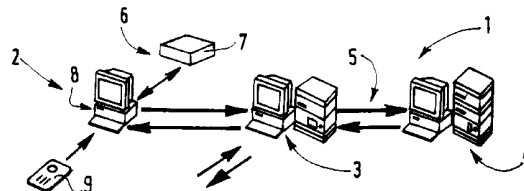
⑦④ Mandataire(s) : BOUJU DERAMBURE BUGNION SA.

⑤④ PROCÉDE, DISPOSITIF, SYSTÈME EN RESEAU ET SUPPORT D'INFORMATIONS CODÉES, AUTOMATIQUES DE CONFIGURATION SÉCURISÉE, DE MÉMORISATION/ACCÈS ET DE CALCUL DE COUT D'APPLICATIONS.

⑤⑦ L'invention concerne notamment un procédé automatique de configuration sécurisée, de mémorisation/accès et de calcul de coût d'utilisation d'applications.

Ce procédé comprend au moins les étapes suivantes :

- d'introduction dans un dispositif (2), d'un support d'informations codées (9) tel que carte à puce;
- de sélection de l'application;
- de mise en oeuvre automatisée d'une routine d'appel apte à lire des paramètres discriminants d'accès à l'application;
- une étape de recherche automatisée dans une base de données de registres; et
- dans l'affirmative, une étape de modification automatisée de fichiers d'initialisation de l'application.



L'invention concerne un procédé automatique de configuration sécurisée d'applications, de mémorisation/accès à des informations relatives à ces applications, et de calcul de coût d'utilisation de telles applications.

Elle concerne également un dispositif automatique de configuration sécurisée de ces applications, de mémorisation/accès à des informations relatives à de telles applications, et de calcul de coût d'utilisation que ces applications, et un système comportant en réseau de tels dispositifs.

L'invention concerne aussi un support d'informations codées, destiné à permettre la lecture et/ou l'inscription de telles informations, dans un tel procédé et/ou dispositif et/ou système.

Le domaine technique de l'invention vise donc notamment le traitement électronique de données, l'enregistrement ou l'indication de fonctionnement de machines, par exemple le temps d'utilisation d'une application par un dispositif tel qu'ordinateur, ainsi que les dispositions à déclenchement pour recevoir ou adresser des transactions.

Concernant le support d'informations codées, des techniques sont citées ici.

Le document FR-A-2.139.258 décrit une carte d'information dite "carte à puce".

Cette carte peut servir de carte de crédit, de carte de compte ou analogues, dans un système commandé par ordinateur.

A cette fin, il est possible d'entrer de manière contrôlable sur la carte, une information en réponse à des ordres donnés par le système d'ordinateur.

La carte comporte des mémoires monolithiques.

5 Une mémoire monolithique comprend une matrice de commutateurs à semi-conducteurs, sur un bloc semi-conducteur dont on peut modifier l'état, de manière à introduire une information binaire en réponse à un signal électrique de déclenchement.

10 Dans une réalisation, il s'agit d'une matrice à diodes opposées monolithiques.

A la fabrication, la mémoire est vide de toute information, et ne contient donc que des zéros.

15 En insérant la carte dans un système de multiplexage, des signaux sont appliqués à la matrice.

L'énergie des signaux modifie les éléments électriques en provoquant leur changement permanent d'état électrique, soit une transformation de zéro en un.

20 Une telle carte peut aussi servir à manipuler des données, par exemple représentant une valeur monétaire. Elle comporte alors un code personnel de sécurité.

25 Le document FR-A-2.374.702 décrit une carte effaçable d'activation et de perception de paiement, pour la distribution de prestations, et une machine fonctionnant avec cette carte, par exemple un taxiphone.

Le coût de la communication par taxiphone étant proportionnel à sa durée et à la distance géographique, la carte équipée d'un circuit électrique intégré constitue une

clé logique d'identification et une matrice de prélèvement de montants correspondant à la prestation fournie.

Des bornes du circuit logique établissent le contact électrique entre la machine et la carte.

5 Dans le domaine des communications par réseaux ou analogues, le document EP-A-008.639 décrit un réseau d'accès pour télécommunications, commandé par carte.

10 Des fonctions centrales sont mises à jour pour extraire des détails d'appels depuis une station locale, et générer un enregistrement de paiement.

Le document EP-A-041.261 décrit un paiement par carte électronique à mémoire programmable, pour un téléphone.

15 Le document WO-A-81.02.655 décrit une authentification pour réseau de transaction en direct, pour permettre l'utilisation d'un service.

20 L'abrégé du document JP-40.37.975 décrit l'exécution de traitements à fonctions multiples, en réponse à une carte unique porteuse sélectivement d'une pluralité de programmes d'applications emmagasinés, par référence à des données d'identification sur la carte.

Lorsqu'une commande est émise par un terminal, le programme correspondant à un code d'identification est exécuté.

25 Concernant les applications, par exemple le document US-A-4.965.066 décrit un traitement de texte.

Tandis que des procédés, dispositifs ou systèmes numériques de traitement d'informations tels que les

ordinateurs personnels dits "PC" ou modulateurs/démodulateurs dits "modems", sont décrits dans de nombreux documents.

5 Le vocabulaire spécifique à ces ordinateurs, est par exemple expliqué dans "Dictionnaire d'informatique", publié par Hermann et édité par Lavoisier.

10 Quant à la traduction en langue française de ce vocabulaire, il est fait référence au "Glossaire des termes officiels de l'informatique", publié par le Journal officiel de la république française, et édité par l'AFNOR.

15 Ainsi, les codes standards américains pour l'échange d'information, dits "ASCII", sont employés sur de nombreuses machines, prévoient des codes sur sept bits sans recommandation de parité, fournissant des profils binaires de caractères.

Le terme "compactage" décrit un procédé de réduction de l'espace mémoire inutilisé.

20 Le terme "compression" signifie un procédé de codage de données afin de tirer parti de redondances, en vue de réduire la taille des données, par exemple pour leur transmission ou leur emmagasinage ou "stockage".

Les techniques connues présentent des inconvénients.

25 Ainsi, elles ne permettent pas d'utiliser un support d'information transportable, tel que carte à puce, afin de personnaliser et de simplifier l'accès à au moins un service ou application accessible à l'aide d'un dispositif et/ou système informatique.

En effet, il est nécessaire pour accéder et utiliser ces applications, de maîtriser certaines connaissances dans le domaine informatique.

5 Outre ces limitations de taille de mémoire, les supports tels que carte à puce, posent des problèmes de lenteur d'accès aux informations qu'ils contiennent.

10 Par ailleurs, un tel support ne permet pas actuellement de mémoriser, préalablement à une première utilisation du support notamment, des procédures d'accès automatisées à des applications d'intérêt pour le porteur autorisé du support, en toute confidentialité et sécurité.

15 Tandis que les techniques actuelles ne proposent pas dans la pratique, de permettre la sauvegarde de telles procédures d'accès sur un support transportable, en toute confidentialité et sécurité, de manière à ce que le porteur d'un support puisse après avoir interrompu son utilisation de l'application, y accéder sans avoir à réitérer ces procédures.

20 Tel est le cas des cartes à puce, du fait des limitations de capacité de mémoire et de temps d'accès évoquées.

25 Par ailleurs, les techniques actuelles ne sont pas satisfaisantes pour assurer de façon simple et économique, un stockage sélectif d'informations et leur transmission à un système de gestion en réseau, en vue de traitements tels que facturation détaillée, analyse commerciale ou analogues.

L'invention vise à résoudre ces inconvénients, notamment.

A cet effet, un premier objet de l'invention est un procédé automatique de configuration sécurisée d'au moins un dispositif tel qu'ordinateur, pour l'exécution d'au moins une application.

5 L'application est par exemple un programme informatique de courrier électronique, de traitement de données tel que tableur, ou analogues.

 Ce procédé comprend au moins les étapes suivantes :

 - une étape d'introduction par un utilisateur dans
10 le dispositif, d'un support d'informations codées tel que carte à puce, ces informations étant aptes à être lues et inscrites par ce dispositif, le support contenant des paramètres discriminants d'accès à l'application ;

 - une étape de sélection par l'utilisateur, de
15 l'application ;

 - un étape de mise en oeuvre automatisée dans le dispositif, d'une routine d'appel apte à lire dans le support d'informations codées, les paramètres discriminants d'accès à l'application ;

20 - une étape de recherche automatisée dans une base de données de registres du dispositif, apte à déterminer en fonction des paramètres discriminants si l'application à configurer est installée dans le dispositif ; et

 dans l'affirmative, une étape de modification
25 automatisée de fichiers d'initialisation de l'application dans le dispositif, par intégration à ces fichiers de paramètres discriminants depuis le support.

 Selon une caractéristique, l'étape de sélection par l'utilisateur, de l'application, comprend une phase
30 préalable de saisie sur le dispositif d'un code confidentiel d'identification par l'utilisateur, sa

comparaison par le dispositif avec une information numérique codée contenue dans le support, et soit :

- une phase de passage à l'étape suivante du procédé de configuration automatisée dans le cas d'une validation du code confidentiel d'identification ;
- une phase d'exécution d'une routine de sécurité dans le cas où le code confidentiel d'identification n'est pas validé.

Selon une autre caractéristique, dans la mesure où l'étape de recherche automatisée s'avère négative et où il est accédé sans configuration automatisée à au moins une application, le procédé effectue une étape de sélection puis de saisie automatisée sur le support, de paramètres discriminants de l'application, afin de permettre une mise en oeuvre automatisée ultérieure de la configuration.

Suivant un exemple de procédé de configuration automatique d'une application de courrier électronique ou "E-mail", les paramètres discriminants de la routine d'appel contiennent notamment :

- le nom de l'utilisateur ;
- son adresse de retour ;
- les coordonnées d'un compte de points de présence de l'utilisateur, dit compte "POP" ;
- les coordonnées d'un serveur de points de présence de l'application, dit serveur "POP" ; et
- éventuellement, les coordonnées d'un serveur d'adressage au protocole de transport de l'application, dit serveur "SMTP".

Suivant une mise en oeuvre d'un tel exemple, l'étape de recherche vise la localisation d'au moins un fichier d'initialisation appelé clé, et lorsque cette clé est présente, l'étape de modification comprend dans les

paramètres discriminants de l'application, des codes d'adresse d'installation que la clé contient.

5 Selon une autre caractéristique, une étape d'effacement automatisée, inverse à l'étape de modification de fichiers d'initialisation de l'application choisie, est exécutée après l'utilisation de l'application, de manière à déconfigurer cette dernière dans le dispositif, en assurant ainsi la confidentialité pour l'utilisateur de l'application, par rapport à d'autres utilisateurs du même
10 dispositif, par exemple ultérieurs.

Selon une mise en oeuvre, l'étape inverse d'effacement, comprend l'effacement des fichiers d'initialisation, et/ou de fichiers produits lors de l'utilisation de l'application.

15 Un second objet de l'invention est un procédé de mémorisation et/ou d'accès dynamique à des informations relatives à de telles que ces applications, sur un support d'informations numériques codées, tel que carte à puce.

20 Ce procédé comporte une étape de compression des informations à mémoriser sur le support à partir d'un dispositif tel qu'ordinateur.

Le procédé conforme à l'invention prévoit au moins deux fonctionnalités, chacune comportant des étapes distinctes, à savoir :

- 25 - une fonctionnalité de compression, par un dispositif tel qu'ordinateur, des informations à enregistrer sur le support ; et
- une fonctionnalité d'accès dynamique, par un dispositif tel qu'ordinateur, à ces informations sur le
30 support.

Pour la compression, une étape prévoit la mise en place d'au moins une table de conversion statique, cette table étant enregistrée sur le dispositif, apte à être lue par celui-ci et comparé avec des informations sur le support.

Dans cette table de conversion statique, sont consignés d'une part des motifs correspondant chacun à une chaîne de caractères à-mêmes d'être répétés.

D'autre part, sont consignés dans la table de conversion statique, une désignation courte codifiée sous une forme numérique comportant un nombre de caractères inférieur à celui du motif correspondant.

Selon une caractéristique, il est préalablement prévu, lors d'une étape de mémorisation préalable, d'enregistrer dans une première section de mémoire du support sans effectuer de compression, l'ensemble d'information à compresser.

Selon une autre caractéristique ultérieure, une étape de localisation du procédé prévoit alors dans le dispositif de passer en revue un ensemble d'informations à compresser, en vue de détecter dans cet ensemble la présence d'au moins une chaîne de caractère reproduisant un motif de la table statique.

Un motif doit présenter plus de trois caractères, les motifs répondant à cette condition étant les seuls à être compressés.

Selon encore une autre caractéristique, lors d'une étape de substitution ultérieure, il est constitué dans le dispositif puis enregistré dans une deuxième section de mémoire du support, un ensemble d'information de premier

niveau de compression, dans lequel chaque motif de la table est remplacé par la désignation courte codifiée correspondante.

5 Dans une mise en oeuvre, cet ensemble d'information de premier niveau de compression, cette désignation comporte un préfixe caractéristique à l'exécution d'une substitution.

10 Selon une caractéristique, il est choisi des caractères non imprimables pour les codes de contrôle, dans le procédé de compression de l'invention et/ou chaque élément d'information est qualifié quant à sa fonction dans l'ensemble d'informations à compresser.

15 Par exemple, certains éléments devant être protégés contre l'effacement et/ou d'autres qualifiés d'auto-démarrables, provoquent l'association à chaque chaîne de caractère ou type d'information, d'un identifiant de qualification apte à définir leur fonction de destination dans l'application.

20 Selon une caractéristique de mise en oeuvre du procédé de compression, les étapes de premier niveau sont suivies d'étapes de deuxième niveau comprenant une étape de consignation d'une désignation courte codifiée sous une forme numérique comportant un nombre de caractères inférieur à celui de chaque motif à compresser, dans une
25 table de conversion dynamique.

Dans une mise en oeuvre, la compression deuxième de niveau comprend les étapes prévoyant les :

- localisation d'une chaîne d'au moins trois caractères dont l'occurrence est supérieure à "n", ici

initialement égal à deux, dans l'ensemble d'informations à compresser ;

- écriture de chaque chaîne localisée dans la table de conversion dynamique dans le dispositif ;

5 - attribution d'un code de substitution ou désignation courte à chaque chaîne écrite dans la table;

- substitution dans l'ensemble d'informations à compresser de la chaîne par la désignation courte correspondante ;

10 - réitération de ces étapes jusqu'à substitution de toutes les chaînes de caractères de l'ensemble d'informations à compresser ; et

- Enregistrement de la table dynamique sur le support tel que carte à puce.

15 Selon une caractéristique, le procédé permettant l'accès dynamique comporte les étapes suivantes :

- une étape de lecture réelle de l'ensemble d'information concerné sur le support tel que carte à puce est effectuée ;

20 - puis, une étape de sauvegarde toujours sur le support de l'information relevée par cette lecture réelle, par exemple suite à des étapes de compression, est effectuée en enregistrant cette information dans une section de mémoire dite tampon ;

25 - ensuite, une étape de décomposition de l'information enregistrée dans la section de mémoire tampon, par exemple par décompression, est effectuée ;

30 - une étape ultérieure prévoit un autre enregistrement de l'information décomposée sur le dispositif, dans une section distincte de mémoire à accès prioritaire à la lecture par le dispositif, de sorte que toute tentative de lecture est orientée vers la section distincte de mémoire à accès prioritaire, et est rendue

immédiate du fait que l'ensemble d'informations est enregistré dans cette section à accès prioritaire sous une forme directement lisible.

5 Selon une caractéristique, lors du retrait du support, une étape prévoit l'invalidation des sections de mémoire tampon et à accès prioritaire, de façon à permettre une mise à jour en cas de changement de support.

10 Selon une autre caractéristique, des étapes relatives à l'accès en écriture, apte à procéder à l'enregistrement des modifications apportées au contenu d'une section du support, avant tout retrait éventuel de celui-ci, comprend une étape telle qu'à chaque accès en écriture, est effectué un accès physique d'enregistrement sur le support.

15 Selon encore une autre caractéristique, une étape de comparaison de l'état instantané de la section de mémoire du support de destination de l'ensemble d'informations, par exemple le contenu d'une section de mémoire tampon, avec l'ensemble d'information à écrire dans
20 le dispositif est effectué ;

 une étape ultérieure prévoit la définition d'un sous ensemble d'informations formé par la différence issue de la comparaison entre l'état instantané de la section de mémoire du support de destination et l'ensemble
25 d'information à écrire ;

 puis, une étape d'enregistrement sur le support de ce sous ensemble exclusivement est exécutée.

 Dans un exemple, l'étape d'enregistrement précède soit :

une étape de mise à jour du contenu de la section de mémoire de destination et dans une section de mémoire tampon ; soit

5 une étape d'invalidation de l'écriture interrompt l'accès aux sections de mémoire de destination et tampon, et provoque vers le dispositif l'émission d'un message d'erreur commandant la mise à jour lors d'une lecture ou écriture ultérieure sur le support.

10 Dans un exemple, une étape de lecture préalable du support est automatiquement provoquée avant l'étape de comparaison, dans le cas où une requête d'écriture est émise sans qu'une étape de lecture ait été effectuée avant la comparaison.

15 Un troisième objet de l'invention est un procédé automatique de calcul de coût d'utilisation d'au moins une application.

Le procédé comprend les étapes prévoyant de :

20 - de détecter, de manière répétitive par exemple à intervalles de temps réguliers de l'ordre de 10 secondes, chaque application en mémoire d'un dispositif tel qu'ordinateur ;

- de définir une durée constante et forfaitaire de facturation, par exemple de l'ordre de 30 secondes, et un taux de facturation de cette durée ;

25 - d'associer une unité de temps de référence propre à chaque application, correspondant à un multiple entier, par exemple compris entre 0 et 3, la durée forfaitaire ;

30 - de mesurer un temps réel d'utilisation de l'application, par exemple égal à "n" unités de temps comptées à intervalles de temps régulier et/ou à l'issue de l'utilisation de l'application ;

- de déterminer par une fonction, un coût égal au taux de facturation multiplié par le nombre d'unités dans le temps réel;

5 - d'enregistrement sélectif de ce coût sur le support d'information, et d'adressage via un réseau d'un message de débit correspondant, à un dispositif serveur de gestion centralisée ; et

- d'opération d'un débit en fonction d'au moins un coût ainsi déterminé.

10 Selon une caractéristique, l'étape de détermination prévoit que :

- l'utilisation d'au moins une application est forfaitaire, voire gratuite ;

15 - lorsque plusieurs applications exécutées en même temps en mémoire du dispositif, celle dont l'unité de référence est la plus élevée, est considérée dominante ;

- lorsqu'au moins une nouvelle application non gratuite est détectée en mémoire, une unité est débitée ;

20 - lorsqu'au moins une application n'est pas initialement associée à une unité de référence, elle est considérée forfaitaire, par exemple d'une unité par minute.

25 Selon une autre caractéristique, le dispositif serveur en réseau dit de gestion d'adressage tient à jour une table de coûts d'application dite table d'administrateur redéfinissable.

30 Selon encore une autre caractéristique, destinée à une application d'exploitation dite "multitâche", pouvant avoir plusieurs sous-applications exécutables simultanément en mémoire, l'étape de détection comprend une phase de comparaison des applications détectées avec liste d'exclusion consignant toutes les applications déjà en mémoire, pour que l'utilisateur ne soit pas débité d'une

manière forfaitaire pour l'utilisation d'applications qu'il n'a pas lancé.

5 Un quatrième objet de l'invention est un dispositif automatique permettant la mise en oeuvre de l'un au moins des procédés évoqués plus haut.

Ce dispositif tel qu'ordinateur de type PC, est apte à mettre en oeuvre au moins un procédé selon l'invention, tel qu'évoqué.

10 Selon une caractéristique, le dispositif comporte un lecteur de support d'informations codées, tel que une carte à puce.

15 Selon une autre caractéristique, le dispositif comporte un modulateur/démodulateur apte à le relier à un réseau, par exemple à un dispositif dit "serveur" de manière à pouvoir lui donner accès à des applications telles que courriers électroniques ou navigation et/ou le dispositif est en libre-service, pour permettre à différents utilisateurs d'employer des applications.

20 Un cinquième objet de l'invention est un système comportant en réseau de tels dispositifs et/ou apte à mettre en oeuvre l'un des procédés évoqués plus haut.

25 Le système comporte au moins deux dispositifs reliés par exemple via un réseau de communication tel que internet, au moins un dispositif étant un serveur de gestion centralisée, notamment destiné à permettre la facturation d'applications exécutées sur un autre dispositif du réseau.

Un sixième objet de l'invention est un support d'informations codées, destiné à permettre la lecture et/ou

l'inscription de telles informations, dans un tel dispositif et/ou système, et/ou selon l'un au moins des procédés évoqués plus haut.

5 Ce support est équipé d'un microprocesseur comportant apte à définir au moins deux sections de mémoire, pour l'enregistrement et l'accès à des informations codées telles que paramètres de configuration d'application, table de conversion ou clé confidentielle d'identification.

10 L'invention ressort mieux de la description détaillée qui suit de modes de réalisation donnés à titre d'exemple.

15 D'une part, cette description se réfère aux annexes 1 à 6, qui comprennent des exemples de modes de réalisation.

L'annexe 1 comprend en exemple 1, un tableau des noms et temps de variables d'entrée dans un procédé de calcul de coût d'applications, selon l'invention.

20 Un exemple 2 sur l'annexe 1 représente un tableau des noms d'applications à lancer, dans un exemple de ce procédé.

Un exemple 3 sur l'annexe 1 représente un tableau définissant des étapes d'un algorithme 1, dans un exemple de ce procédé.

25 Un exemple 4, sur l'annexe 3, représente un tableau définissant des étapes d'un algorithme 2.

Comme les exemples 1 à 4, les parties 1 et 2 des annexes 4 à 6, concernent aussi un procédé de calcul de coût d'applications.

5 Ces parties illustrent un exemple de programme source de calcul de coût d'application, en langage de programmation Pascal.

Ces sources illustrent des exemples d'algorithmes généraux dits de principe, pour un tel procédé.

10 Sur l'annexe 1 encore, un exemple 5 concerne un procédé de compression d'informations numériques codées et d'accès dynamique à de telles informations, sur un support tel que carte à puce.

L'exemple 5 comporte trois tableaux, respectivement désignés en 5a, 5b et 5c.

15 Le tableau 5a définit un ensemble d'informations avant compression.

Le tableau 5b présente un extrait de table de conversion de motifs en vue de la compression, et leur code de substitution respectif.

20 Et le tableau 5c représente un ensemble d'informations dans lequel une compression a été effectuée par substitution de motifs, avec un caractère de marquage.

25 Dans les tableaux 5a et 5c, les cases hachurées représentent des espaces mémoires inutilisés, tandis que les cases noircies représentent le gain de compression.

Et le caractère 0, à la ligne 3 colonne 2 par exemple, constitue un délimiteur d'information.

Les annexes 2 et 3 comprennent un exemple 6, avec des tableaux 6a à 6f, illustrant le contenu de mémoires d'un support tel que carte à puce, à différentes étapes d'un exemple de procédé de compression d'information de premier niveau.

Dans ces tableaux les caractères 0 et les cases hachurées ont la même signification que dans l'exemple 5.

Il en va de même dans l'exemple 7.

Le tableau 6a représente le contenu de trois sections de mémoire d'un tel support, avant la compression.

Le tableau 6b représente le contenu des trois sections de mémoire, après une étape de tentative de lecture de ce contenu, de copie d'une première section CP dans une deuxième section MT1, et un remplissage d'une troisième section MT2 de mémoire tampon par un algorithme de décompression.

Le tableau de l'exemple 6c représente une étape de modification d'un ensemble d'informations, dans la section de mémoire tampon MT2.

Les cases grisées (lignes 9 et 10, et les cinq premières cases de la ligne 10) représentent sur le tableau 6C, un ensemble de codes modifiés dans section MT2.

Le tableau de l'exemple 6d représente une étape d'application d'un algorithme de compression sur la section MT2 par copie d'un résultat dans la section MT1, qui est alors désignée alors en MT1bis.

Le tableau de l'exemple 6e représente une étape de localisation de différences entre les sections de mémoires MT1 et MT1bis.

5 Dans ce tableau 6e, les trois dernières cases de la ligne 6, la ligne 7 et les six premières cases de la ligne 8 dans la section MT1, ainsi que les mêmes cases dans la section MT1bis, illustrent les différences sélectionnées au sein des sections MT1 et la nouvelle section MT1bis.

10 Sur l'annexe 3, le tableau de l'exemple 6f représente après une étape d'effacement du contenu du support correspondant aux différences sélectionnées dans la section MT1, puis d'écriture de nouvelles informations disponibles dans les cases correspondantes de la section MT1bis, le contenu des sections CP, MT1 et MT2.

15 Notons que le code Ix désigne un code de qualification de l'information sur huit bits, préalablement désigné par I1, I2 ou I3.

20 L'annexe 3 comprend un exemple 7, avec des tableaux 7a, 7b et 7c, illustrant le contenu de mémoires d'un support tel que carte à puce, à différentes étapes d'un exemple de procédé de compression d'information de deuxième niveau.

25 Dans cet exemple, les mêmes informations que dans l'exemple 6 ont été choisies, pour plus de simplicité. Tandis que la compression de premier niveau a été effectuée préalablement.

Le tableau 7a de l'exemple 7 illustre l'état des informations qui en découle.

Le tableau 7b illustre le résultat d'une étape de recherche de chaînes, dans l'exemple d'au moins trois caractères, répétées au moins deux fois dans le contenu d'une section de mémoire que l'on souhaite compresser.

5 Et le tableau 7c représente les résultats d'une étape de substitution de ces chaînes par un code de substitution illustré sur le tableau 7b.

10 Dans ce tableau, les seize dernières cases de la quatrième ligne représentent les modifications apportées à la section de mémoire.

D'autre part, la description détaillée de l'invention qui suit, se réfère aux dessins annexés, donnés uniquement à titre d'exemple.

15 Dans ces dessins, la figure 1 représente un système en réseau comprenant des dispositifs, ici des ordinateurs de type PC, aptes à mettre en oeuvre au moins un procédé selon l'invention, tel qu'évoqué.

20 Les figures 2 à 4 sont des graphiques illustrant, dans un procédé de compression et d'accès dynamique, les gains obtenus.

Plus précisément, la figure 2 illustre le gain obtenu en fonction de la longueur et du nombre de répétitions, avec en abscisses la longueur (L) et en ordonnées le gain en pourcents (%).

25 La figure 3 illustre aussi le gain obtenu en fonction de la longueur et du nombre de répétitions, avec en abscisses la longueur (L) et en ordonnées le gain en pourcents (%).

La figure 4 est un graphique qui illustre la longueur minimale d'une chaîne en fonction du nombre de répétitions, dans un procédé de compression et d'accès dynamique, avec en ordonnées la longueur (L) et en abscisses le nombre de répétitions (N).

La figure 5 est un organigramme logique qui illustre un exemple d'algorithme dit ALGO 1 dans un procédé de calcul de coûts d'applications.

Et la figure 6 est un organigramme logique qui illustre un autre exemple d'algorithme dit ALGO 2 dans un procédé de calcul de coûts d'applications.

Sur la figure 1 un système informatique 1 en réseau comprenant plusieurs dispositifs 2, 3 et 4 tels qu'ordinateurs de type PC, sont représentés.

Ces système 1 et dispositifs 2 sont aptes à mettre en oeuvre au moins un procédé selon l'invention, tel qu'évoqué.

Le système 1 comporte aussi un réseau 5 de connexion des dispositifs 2 à 4, par exemple un réseau dédié.

Un réseau 6, par exemple de communication tel que internet, et pouvant comprendre le réseau 5 est aussi représenté.

Le dispositif 2 comporte un modulateur/démodulateur 7 qui le relie au réseau 6.

Le dispositif 3 est un dispositif dit "serveur", connecté au dispositif 2 de manière à pouvoir lui donner

accès à des applications telles que courriers électronique ou navigation, notamment.

5 D'autres dispositifs comparables au dispositif 2 sont connectés à ce serveur 3, qui fait donc office de serveur d'adressage.

Tandis que le dispositif 4 est un serveur de gestion centralisée, notamment destiné à permettre la facturation d'applications exécutées sur le dispositif 2, ou sur d'autres dispositifs similaires non représentés.

10 Typiquement, le dispositif 2 est en libre-service, pour permettre à différents utilisateurs d'employer des applications.

Au moins un dispositif 2 est dans un exemple la propriété de ou loué à l'utilisateur.

15 Afin de pouvoir personnaliser le dispositif 2 en fonction de chaque utilisateur, et assurer une facturation proportionnelle à leur utilisation d'applications, le dispositif 2 comporte un lecteur 8 de support d'informations codées.

20 Sur la figure 1, ce support est une carte à puce désignée en 9.

Le support 9 d'informations codées, est comme on le comprend plus loin, destiné à permettre la lecture et/ou l'inscription d'informations.

25 Ce support 9 est équipé d'un microprocesseur apte à définir au moins deux sections de mémoire par exemple MT1, MT1bis, MT2, pour l'enregistrement et l'accès à des informations codées telles que paramètres de configuration

d'application, table de conversion ou clé confidentielle d'identification.

5 Dans des exemples, les sections de mémoire ainsi que les informations enregistrées sont préalablement inscrites sur le dispositif 8, et selon les cas copiées sur le support 9.

Le dispositif 2 permet ainsi la mise en oeuvre d'un procédé de configuration automatique sécurisée.

10 Le but de ce procédé est de permettre l'accès simplifié à des applications.

Ces applications sont telles que logiciels de courrier électronique ou "E-mail", navigation sur des réseaux globaux comme internet.

15 Mais sont aussi visées ici des traitement de texte, tableur, questionnaire de base de données, programme de dessin ou animation, et analogues.

Ce procédé prévoit une routine d'appel apte à présenter une structure contenant des paramètres discriminants de l'application choisie.

20 Dans l'exemple d'une application de courrier électronique ou "E-mail", la routine d'appel contient :

- le nom de l'utilisateur ;
- son adresse de retour ;
- le compte dit "POP" ;
- 25 - le serveur dit "POP" ; et
- le serveur dit "SMTP".

Le procédé de configuration effectue pour chacune des applications à gérer, une recherche dans une base de

données de registres, si l'application à configurer est installée sur le dispositif 2, ou accessible via celui-ci depuis le serveur 3 ou le modem 7.

5 Dans l'affirmative, le procédé effectue alors une étape de modification de fichiers d'initialisation de l'application choisie, dits fichiers "INI".

10 Dans la mesure où la recherche s'avère négative, le procédé effectue la saisie automatisée ou entrée de la routine d'appel de l'application, dans la base de données de registres.

 Une fonction inverse à l'étape de modification de fichiers d'initialisation de l'application choisie, permet de déconfigurer et d'effacer dans le dispositif recevant le support tel que carte à puce.

15 Cela assure ainsi de conserver la confidentialité pour l'utilisateur de l'application, par rapport à des utilisateurs ultérieurs du même dispositif.

20 Dans un premier exemple de courrier électronique appelé Eudora (R), la configuration prévoit la recherche dans la base de registre d'un chemin d'accès à un fichier d'initialisation pointé par un sous-fichier de la base dit clé "QUALCOM".

 Lorsque cette clé est présente, les codes d'adresse dits "PATH" d'installation que la clé contient sont saisis.

25 Ensuite, des informations sélectionnées dans les codes d'adresse et/ou clé sont copiés notamment dans le support tel que carte à puce.

Pour la déconfiguration, dans les fichiers d'initialisation sont retirés, c'est-à-dire effacés, les entrées. Puis est effectué l'effacement des fichiers produits lors de l'utilisation de l'application.

5 Dans un deuxième exemple de courrier électronique appelé Netscape (R), la configuration prévoit la recherche dans la base de registre d'un fichier d'initialisation contenant les codes "PATH" d'installation

Si ces codes ou clé existent, elles sont saisies.

10 Ensuite, des informations sélectionnées dans les codes d'adresse et/ou clé sont copiés.

Pour la déconfiguration, il est procédé comme exposé plus haut.

15 Dans un exemple d'utilisation sur le réseau internet pour le courrier électronique, le procédé suit notamment les étapes suivantes.

La configuration prévoit la recherche dans la base de registre d'un fichier d'initialisation appelé "clé INTERNET MAIL".

20 Lorsque cette clé est présente, les racines de codes d'adresse dits "PATH" d'installation que la clé contient sont spécifiés dans un répertoire temporaire d'une application d'exploitation à interface graphique telle que Windows (R) dans sa version 1995.

25 Ensuite, des informations sélectionnées dans les codes d'adresse et/ou clé sont copiés dans les différentes entrées de la base de registre, depuis le support tel que carte à puce.

Pour la déconfiguration, dans les fichiers d'initialisation sont retirés, c'est-à-dire effacés, les entrées. Puis est effectué l'effacement des fichiers produits lors de l'utilisation de l'application de courrier électronique par internet.

Il va de soi que pour d'autres applications, telles que traitement de texte, tableur, gestionnaire de base de données, ou analogues, des étapes similaires peuvent être prévues, pour simplifier et rendre plus rapide un accès à ces applications, et leur configuration notamment en fonction de préférences ou accès antérieurs de l'utilisateur.

Un procédé généralement dit de mémorisation d'informations sur un support tel que carte à puce est maintenant décrit.

Les cartes à puces ou à microprocesseurs actuellement répandues sur le marché ne possèdent pas une taille ou capacité de mémoire suffisante pour enregistrer par exemple des paramètres discriminants nécessaires à la configuration et/ou à la personnalisation d'applications, comme cela vient d'être exposé.

Il a aussi été souligné qu'avec les cartes actuellement répandues sur le marché, les temps d'accès, notamment en écriture, sont relativement longs.

De manière antagoniste, le volume des fichiers mis en oeuvre pour le fonctionnement des applications est en constante augmentation.

Il est donc de plus en plus difficile de pouvoir enregistrer sur un support tel que carte à puce, des

ensembles de données numérisées permettant l'exécution de procédés tels qu'exposés.

5 Les techniques actuelles de compression ou d'optimisation s'avèrent pour ce faire, insatisfaisantes dans la pratique.

Pour concilier ces antagonismes, le procédé conforme à l'invention prévoit deux fonctionnalités, chacune comportant des étapes distinctes, à savoir:

10 - une fonctionnalité de compression, par un dispositif tel qu'ordinateur, des informations à enregistrer sur le support; et

- une fonctionnalité d'accès dynamique, par un dispositif tel qu'ordinateur, à ces informations sur le support.

15 La première vise à minimiser la taille de l'information à enregistrer sur le support, alors que la deuxième optimise les accès sur le support, en limitant la quantité d'informations à écrire et/ou effacer, par exemple pour exécuter l'un au moins des procédés évoqués ici.

20 Le but de la compression est de transcrire sous forme codifiée une partie d'information, typiquement une série de caractères.

25 A cette fin, il est mis en place au moins une table de conversion statique. Cette table est enregistrée sur le dispositif 8 et apte à être lue par celui-ci.

Dans un exemple, la table statique est directement enregistrée sur le support 9.

Le terme statique signifie ici que la table de conversion n'est pas appelée à subir des évolutions en fonction des informations à compresser.

5 Dans cette table de conversion statique, sont consignés d'une part des motifs correspondant chacun à une chaîne de caractères à-mêmes d'être répétés.

Cette répétition, intervient soit au sein d'un même ensemble d'informations, tel que fichier.

10 Soit il s'agit de chaînes de caractères communément employés dans les applications, et à occurrence fréquente.

Il s'agit typiquement d'instructions, de variables, de constantes, d'adresses, d'extensions, de codes normalisés ou analogues.

15 D'autre part, sont consignés dans la table de conversion statique, une désignation courte codifiée sous une forme numérique comportant un nombre de caractères inférieur à celui du motif correspondant.

20 Il est préalablement prévu, lors d'une étape de mémorisation préalable, d'enregistrer dans une première section de mémoire sans effectuer de compression, l'ensemble d'information à compresser.

25 Ultérieurement, une étape de localisation du procédé prévoit alors dans le dispositif de passer en revue un ensemble d'informations à compresser, en vue de détecter dans cet ensemble la présence d'au moins une chaîne de caractère reproduisant un motif de la table statique.

Lors d'une étape de substitution ultérieure, il est constitué dans le dispositif puis enregistré dans une

deuxième section de mémoire, un ensemble d'information de premier niveau de compression, dans lequel chaque motif de la table est remplacé par la désignation courte codifiée correspondante.

5 Dans cet ensemble d'information de premier niveau de compression, afin d'éviter une confusion à la lecture par le dispositif entre des caractères extérieur à et compris dans une désignation courte, cette dernière comporte un préfixe caractéristique à l'exécution d'une
10 substitution.

On comprend déjà qu'un motif doit présenter plus de trois caractères pour que la compression présente un intérêt. Donc, les motifs répondant à cette condition seront en général les seuls à être compressés.

15 Afin d'implémenter le procédé de compression, citons ici des conventions actuellement en vigueur :

- une adresse est représentée par une chaîne de caractères ;
- dans un fichier écrit dans un langage de
20 programmation tel que C, une chaîne de caractères est une succession de nombre entier codé sur un octet, soit huit bits ;
- chaque entier représente un caractères ou code de contrôle ;
- 25 - un fichier ou ensemble d'informations est lu selon les conventions ASCII évoquées plus haut en sept ou huit bits.

A partir de ces bases, il est en général choisi des caractères non imprimables pour les codes de contrôle, dans
30 le procédé de compression de l'invention.

Précisons aussi que chaque élément d'information doit être qualifié quant à sa fonction dans l'ensemble d'informations à compresser.

5 Par exemple, certains éléments doivent être protégés contre l'effacement. D'autres sont qualifiés d'auto-démarrables.

10 Ceci implique donc l'association à chaque chaîne de caractère ou type d'information, un identifiant de qualification apte à définir leur fonction de destination dans l'application.

Lors de l'étape de substitution, un identifiant de qualification est associé sur le support à chaque motif compressé, afin d'éviter à la lecture de l'ensemble d'informations le contenant, une interprétation erronée.

15 Suivant le procédé de compression, il est choisi une syntaxe d'information telle que dans l'exemple qui suit :

Ensemble d'information= <Liste d'info>

20 Liste d'info = (code Qualificatif)
<Information>(Code Fin de
Chaîne)<Liste d'info>I f

Information = <Elément de
Base><Information>I f

25 Elément de Base = (Code de Substitution)
(Code d'équivalence) I
(Caractère Imprimable)

Dans cet exemple:

(code Qualificatif) = Valeur comprise entre 01_{HEX}
et FE_{HEX}

(Code Fin de Chaîne) = Valeur égale à 00_{HEX}

(Code de Substitution)= Valeur égale à 01_{HEX}

5 (Code d'équivalence) = Valeur comprise entre 01_{HEX}
et FE_{HEX};

(Caractère Imprimable)= Valeur comprise entre 20_{HEX}
et FE_{HEX}.

10 Le caractère "I" correspond au "ou" logique, et "ε"
au vide épsilon.

Rappelons ici que le procédé de compression, qui
suit par exemple un algorithme, permet de réaliser une
substitution de motifs contenus dans la table de conversion
statique, par une désignation courte codifiée dite aussi
15 code de substitution.

Dès qu'une occurrence est localisée, la substitution
est effectuée.

Suivant une mise en oeuvre du procédé de
compression, les étapes de premier niveau sont suivies
20 d'étapes de deuxième niveau.

Dans ces étapes, une désignation courte codifiée
sous une forme numérique comportant un nombre de caractères
inférieur à celui de chaque motif à compresser est
consignée dans une table de conversion dynamique et non pas
25 statique, comme dans la compression de premier niveau.

En synthèse, la compression deuxième de niveau
comprend les étapes prévoyant de :

- localisation d'une chaîne d'au moins trois caractères dont l'occurrence est supérieure à "n", ici initialement égal à deux, dans l'ensemble d'informations à compresser ;
- 5 - écriture de chaque chaîne localisée dans la table de conversion dynamique dans le dispositif ;
- attribution d'un code de substitution ou désignation courte à chaque chaîne écrite dans la table ;
- substitution dans l'ensemble d'informations à
- 10 compresser de la chaîne par la désignation courte correspondante ;
- réitération de ces étapes jusqu'à substitution de toutes les chaînes de caractères de l'ensemble d'informations à compresser ; et
- 15 - Enregistrement de la table dynamique sur le support tel que carte à puce.

Comme dans le premier niveau de compression, la première désignation courte comporte par exemple le code de substitution 1, et la suivante 2, etc.

- 20 Des délimitateurs de fonction sont prévus en tête d'un motif substitué, par exemple "0x02_{HEX}".

De manière similaire, sur le support la table de conversion est dans cet exemple séparée des autres informations qu'elle contient, par une marque telle que

25 deux codes de type "0xFF".

Les étapes du procédé permettant l'accès dynamique sont maintenant décrites.

Le but de ces étapes est de minimiser les échanges d'informations entre le support tel que carte et le

dispositif à partir duquel est exécutée au moins une application.

Il s'agit donc de mettre en place une sorte de mémoire cache.

5 Un tel accès dynamique vise un ensemble d'informations, tel qu'un fichier entier, des chaînes de caractères ou des motifs d'information.

Le procédé comporte, pour l'accès dynamique, les étapes suivantes.

10 Une étape de lecture réelle de l'ensemble d'information concerné sur le support tel que carte à puce est effectuée.

15 Puis, une étape de sauvegarde de l'information relevée par cette lecture réelle, par exemple suite à des étapes de compression, est effectuée en enregistrant cette information dans une section de mémoire dite tampon.

Ensuite, une étape de décomposition de l'information enregistrée dans la section de mémoire tampon, par exemple par décompression, est effectuée.

20 Une étape ultérieure prévoit un autre enregistrement de l'information décomposée dans une section distincte de mémoire à accès prioritaire à la lecture par le dispositif.

25 Ainsi, toute tentative de lecture est orientée vers la section distincte de mémoire à accès prioritaire, et est rendue immédiate du fait que l'ensemble d'informations est enregistré dans cette section à accès prioritaire sous une forme directement lisible.

Lors du retrait du support, une étape automatique prévoit l'invalidation des sections de mémoire tampon et à accès prioritaire.

5 Cela permet notamment la confidentialité en cas de changement de support.

Après avoir décrit les étapes relatives à l'écriture sur le support, expliquons les étapes relatives à l'accès en écriture.

10 Ce dernier se différencie de l'écriture, par la nécessité de procéder à l'enregistrement des modifications apportées au contenu d'une section du support, avant tout retrait éventuel de celui-ci.

Ceci implique qu'à chaque accès en écriture, est effectuée un accès physique d'enregistrement sur le support.

15 Une étape de comparaison de l'état instantané de la section de mémoire du support de destination de l'ensemble d'informations, par exemple le contenu d'une section de mémoire tampon, avec l'ensemble d'information à écrire dans le dispositif est effectué.

20 Une étape ultérieure prévoit la définition d'un sous ensemble d'informations formé par la différence issue de la comparaison entre l'état instantané de la section de mémoire du support de destination et l'ensemble d'information à écrire.

25 Puis, une étape d'enregistrement sur le support de ce sous ensemble exclusivement est exécutée.

Dans un exemple, l'étape d'enregistrement précède soit :

une étape de mise à jour du contenu de la section de mémoire de destination et dans une section de mémoire tampon; soit

5 une étape d'invalidation de l'écriture interrompt l'accès aux sections de mémoire de destination et tampon, et provoque vers le dispositif l'émission d'un message d'erreur commandant la mise à jour lors d'une lecture ou écriture ultérieure sur le support.

10 Dans un exemple, une étape de lecture préalable du support est automatiquement provoquée avant l'étape de comparaison, dans le cas où une requête d'écriture est émise sans qu'une étape de lecture ait été effectuée avant la comparaison.

15 Cela permet une mise à jour systématique des sections de mémoire de destination et tampon, avant une écriture.

20 Sur l'annexe 1, est illustré un exemple 5 de procédé de compression d'informations numériques codées et d'accès dynamique à de telles informations, sur un support tel que carte à puce.

Le tableau 5a définit un ensemble d'informations avant compression.

25 Le tableau 5b présente un extrait de table de conversion de motifs en vue de la compression, et leur code de substitution respectif.

On constate en effet que les ensembles d'information comportent des chaînes de caractères occurrents.

Par exemple dans le cas d'une navigation sur un réseau tel qu'internet, on constate :

- Une caractérisation des informations de réseau de transmission :

5 . les adresses de sites internet, commencent par la chaîne de caractères "http://" ;

. les adresses de courrier E-mail, commencent par la chaîne de caractères "mailto:" ;

10 . les adresses de sites de réseaux à protocole de transfert de fichiers dits "FTP", commencent par la chaîne de caractères "ftp://" ;

. les adresses de forums de discussion ou "news group", commencent par la chaîne de caractères "news:" ;

15 - Une caractérisation de la localisation géographique des fournisseurs d'information sur réseau de transmission, notamment selon leur pays d'origine :

. ".fr" pour la France ;

. ".fi" pour la Finlande ;

. ".uk" pour la Grande-bretagne ;

20 . ".jp" pour le Japon ;

. ".au" pour l'Australie, notamment.

- Une caractérisation de la société de fournisseurs d'information sur réseau de transmission.

25 - Une caractérisation de la catégorie d'entité du fournisseur d'information sur réseau de transmission :

. ".edu" pour les établissements d'enseignement ;

. ".com" pour les entités à but lucratif ou commercial ;

30 . ".gov" pour les entités gouvernementales, notamment.

Ces motifs se retrouvent pour certains dans l'exemple du tableau de conversion statique 5a de l'annexe 1.

Et le tableau 5c représente un ensemble d'informations dans lequel une compression à été effectuée par substitution de motifs, avec un caractère de marquage.

5 Les annexes 2 et 3 comprennent un exemple 6, avec des tableaux 6a à 6f, illustrant le contenu de mémoires d'un support tel que carte à puce, à différentes étapes d'un exemple de procédé de compression d'information de premier niveau.

10 Le tableau 6a représente le contenu de trois sections de mémoire d'un tel support, avant la compression.

On a vu que CP désigne une section d'inscription initiale, MT1 ou MT1bis des sections de mémoire tampon, et MT2 une section de mémoire dite à accès prioritaire.

15 Le tableau 6b représente le contenu des trois sections de mémoire, après une étape de tentative de lecture de ce contenu, de copie d'une première section CP dans une deuxième section MT1, et un remplissage d'une troisième section MT2 de mémoire tampon par un algorithme de décompression.

20 Alors, les informations ou données disponibles le sont en accès direct et instantané pour toutes les lectures ultérieures du support, avant un autre enregistrement.

25 Le tableau de l'exemple 6c représente une étape de modification d'un ensemble d'informations, dans la section de mémoire tampon MT2.

Ici, l'information "<http://www.icard.net/main.html>" est modifiée en "<http://www.icard.com/pages/index.html>" dans la section de mémoire tampon MT2.

Le tableau de l'exemple 6d représente une étape d'application d'un algorithme de compression sur la section MT2 par copie d'un résultat dans la section MT1, qui est alors désignée alors en MT1bis.

5 L'étape de comparaison est ainsi effectuée.

Dans un exemple, la mémoire est allouée dynamiquement, et qu'elle sera restituée après son utilisation provisoire dynamique.

10 Le tableau de l'exemple 6e représente une étape de localisation de différences entre les sections de mémoires MT1 et MT1bis.

Cette étape est effectuée sous la commande d'un algorithme.

15 Dans ce tableau 6e, les différences sélectionnées au sein des sections MT1 et la nouvelle section MT1bis sont illustrées.

20 Sur l'annexe 3, le tableau de l'exemple 6f représente après une étape d'effacement du contenu du support correspondant aux différences sélectionnées dans la section MT1, puis d'écriture de nouvelles informations disponibles dans les cases correspondantes de la section MT1bis, le contenu des sections CP, MT1 et MT2.

25 L'annexe 3 comprend un exemple 7, avec des tableaux 7a, 7b et 7c, illustrant le contenu de mémoires d'un support tel que carte à puce, à différentes étapes d'un exemple de procédé de compression d'information de deuxième niveau.

Dans cet exemple, la compression de premier niveau a été effectuée préalablement.

Le tableau 7a de l'exemple 7 illustre l'état des informations qui en découle.

5 Le tableau 7b illustre le résultat d'une étape de recherche de chaînes, dans l'exemple d'au moins trois caractères, répétées au moins deux fois dans le contenu d'une section de mémoire que l'on souhaite compresser.

10 Et le tableau 7c représente les résultats d'une étape de substitution de ces chaînes par un code de substitution illustré sur le tableau 7b.

En se reportant aux figures 2 à 4, sont illustrés les gains obtenus de volume utile de mémoire avec le procédé de compression décrit.

15 Le procédé de calcul du coût des applications est maintenant décrit.

Le but de ce calcul du coût des applications est de détecter les applications en mémoire et de leur associer un coût fonction du temps, entre chaque débit d'unité.

20 Les règles à respecter sont dans les exemples telles que :

- Une application peut être gratuite.
- Si plusieurs applications sont exécutées en même temps en mémoire du dispositif, la plus consommatrice en

25 unité est dominante sur le réglage de la vitesse de débit des unités du compteur.

- Lorsqu'une nouvelle application non gratuite est détectée en mémoire, le compteur est débité d'une unité.

- Lorsqu'une application n'est pas référencée dans la table associant chaque applications a un prix, elle est facturée sur un tarif forfaitaire, par exemple d'une unité par minute.

5 Dans le système 1, un dispositif 4 en réseau dit de gestion tient à jour une table associant à chaque application un coût que l'administrateur du PC peut redéfinir.

10 Un exemple 1 de ce tableau est représenté sur l'annexe 1.

Le temps minimum entre deux débits d'unité est dans l'exemple de 30 secondes.

Seules les applications lancées par l'utilisateur vont être facturées.

15 Le fonctionnement de calcul du coût des applications prévoit de mettre en oeuvre plusieurs logiques opérationnelles :

- Le repérage des applications ou "process" en mémoire.

20 - Les techniques retenues pour connaître à tout moment l'état de la mémoire de l'application opérationnelle, par exemple à interface graphique, par exemple Windows (R), qui sont les suivantes :

25 1) Une temporisation ou "timer" d'une vitesse d'horloge dite "granulaire" de 10 secondes, lance une fonction de vérification du contenu de la mémoire.

30 2) Une autre version prévoit pour non plus de scruter ou "scanner" en permanence la mémoire, mais d'être alerté à chaque démarrage de processus par l'application d'exploitation du dispositif 2.

La première technique s'appuie sur une fonctionnalité de cette application d'exploitation par exemple désignée par "API ToolHelp32".

5 La deuxième serait d'utiliser les fonctionnalités d'accroche dites "HOOK" du type parfois désignées dans des fichiers d'interface de coeur dits "SHELL".

Cette dernière étant beaucoup moins consommatrice de temps machine n'a pas été utilisée car elle oblige le programme, faisant appel à ce HOOK particulier, à gérer
10 toutes les réductions de fenêtre.

Ce HOOK est utilisé par exemple par la barre de tâche d'une application d'exploitation comme Windows (R) dans sa version de 1995.

15 Le deuxième logique opérationnelle prévoit la maintenance d'une liste des coûts.

Cette technique permettant de savoir combien coûte une application, repose sur une liste contenant le nom mémoire de l'application et son coût. Dans les exemples, on prévoit notamment :

20 - pour une application de navigation "NAVIG.EXE" correspond un temps 1 (1 pour 1 X 30 Sec.) ;
- pour une application de carnet portable "NOTESX.EXE" correspond un temps infini: (infini pour gratuit).

25 L'exemple 1 apporte un cas de traitement de texte "TEXTEUR.EXE", et de courrier via internet "INTERNET MAIL".

La troisième logique opérationnelle gère la maintenance d'une liste des applications chargées avant l'exécution du programme, comme sur l'exemple 2, où

"SERVER.EXE" est une application de connexion à un serveur tel que le dispositif 3, et "SYSTER.EXE" est l'application d'exploitation.

5 Une application d'exploitation dite "multitâche" telle que Windows (R) dans sa version 1995, pouvant avoir plusieurs applications exécutables simultanément en mémoire, pour que l'utilisateur ne soit pas débité d'une manière forfaitaire pour l'utilisation d'applications qu'il n'a pas lancé, il est prévu de mettre dans une liste
10 d'exclusion, toutes les applications déjà en mémoire.

On comprend que l'exemple 1 est un tableau des noms et temps de variables d'entrée dans un procédé de calcul de coût d'applications, selon l'invention.

15 Tandis que l'exemple 2 sur l'annexe 1 représente un tableau des noms d'applications déjà lancées, dans un exemple de ce procédé.

Un exemple 3 sur l'annexe 1 représente un tableau définissant des étapes d'un algorithme 1, dans un exemple de ce procédé.

20 Cet algorithme 1 dit ALGO1 est illustré sur la figure 5, à rapprocher du tableau de l'exemple 3.

Sur cette figure, les cases désignées par les références numériques à gauche, représentent les fonctionnalités ou étapes à droite :

- 25
- référence 10 = début de l'algorithme;
 - référence 11 = étape "INIT 1" de remise à l'infini du temps ;
 - référence 12 = étape "FONCTION 1" de NOM= suivant sur la liste des applications ou "processus" actifs ;

- référence 13 = étape "TEST 1" définie par NOM = 0
?";

- référence 14 = étape intervenant si la réponse en
13 est "Y" pour "oui", de fin de l'algorithme;

5 - référence 15 = étape intervenant si la réponse en
13 est "N" pour "non", de passage à une étape ultérieure de
l'algorithme. Cette étape interroge par "NOM=EXPLORER.EXE
?" ;

- référence 16 = étape intervenant si la réponse
10 en 15 est "Y" pour "oui", de passage à un autre algorithme
de deuxième niveau dit "ALGO2" ;

- référence 17 = étape intervenant si la réponse en
15 est "N" pour "non", de passage à une étape ultérieure de
l'algorithme. Cette étape interroge par "NOM appartient à
15 LISTE_NOM_TEMPS ?" ;

- référence 18 = étape "TEST 6" intervenant si la
réponse en 17 est "Y" pour "oui", de passage à une étape
ultérieure de l'algorithme. Cette étape interroge par
"TEMPS<TEMPS de LISTE_NOM_TEMPS à l'index NOM" ;

20 - référence 19 = étape "TEST 4" intervenant si la
réponse en 17 est "N" pour "non", de passage à une étape
ultérieure de l'algorithme. Cette étape interroge par "NOM
appartient à LISTE_NOM ?" ;

- référence 20 = étape "FONCTION 2" intervenant
25 si la réponse en 18 est "N" pour "non", de passage à une
étape ultérieure de l'algorithme. Cette étape interroge par
"TEMPS = TEMPS de LISTE_NOM_TEMPS à l'index NOM" ;

- référence 21 = étape "TEST 5" intervenant si la
réponse en 19 est "N" pour "non", de passage à une étape
30 ultérieure de l'algorithme. Cette étape interroge par
"TEMPS > 2" ;

- référence 22 = étape "FONCTION 3" intervenant si
la réponse en 21 est "Y" pour "oui", de passage à une étape

ultérieure de l'algorithme. Cette étape interroge par "TEMPS = 2".

Le but de cet algorithme ALGO 1 est :

- de comparer avec deux listes prédéfinies, les applications en mémoire.

Une liste dont un exemple 1 est représenté sur l'annexe 1, comprend toutes ces applications avec leurs coûts associés définies par l'utilisateur de l'application.

Une liste dont un exemple 2 est représenté sur l'annexe 1, comprend toutes les applications présentes en mémoire avant le lancement de l'application.

En effectuant cette comparaison, une variable est définie pour déterminer le temps qui doit s'écouler entre le débit de deux unités sur le support d'information tel que carte à puce.

Variables d'entrées :

LISTE_NOM_TEMPS : liste de structures comprenant les noms et les temps N qui leurs sont associés. Son contenu est défini par l'administrateur de réseau, par exemple à l'aide d'un support 9 destiné à cette fin. Ce support 9 dit "dédié" est inséré dans le dispositif 2, pour toute nouvelle définition.

LISTE_NOM : liste comprenant les noms des applications lancées avant l'application.

Variable de sortie :

TEMPS : valeur entière définissant combien de paquets de 30 secondes doivent s'écouler entre 2 débits d'unités.

Variables locales à l'algorithme :

NOM : Contenant à chaque itération le nom d'un des processus actifs, jusqu'au moment où tous ont été listés.

5 De manière similaire, rapprochons maintenant la figure 6 de l'exemple 4 de l'annexe 3. Leur enchaînement et les interrogations ressortent de ces figure et exemple.

Les références numériques identiques, désignent des fonctionnalités ou étapes similaires.

10 Ainsi, la référence 23 est une étape "FONCTION 4" intervenant si la réponse en 21 est "N" pour "non", de passage à une étape ultérieure de l'algorithme. Cette étape interroge par "TEMPS = TEMPS de LISTE_NOM_TEMPS à l'index NOM_THREAD".

15 Sur la figure 6, est illustré un algorithme dit "ALGO2".

Le but de cet algorithme est de permettre que l'explorer de WINDOWS 95 soit considéré en tant que conteneur de données d'évaluation de la facturation, dites "ACTIVEX".

20 Pour pouvoir facturer ces ACTIVEX dans une application telle que courrier Internet Mail, nous devons repérer les éventuels informations de dites "THREAD", que l'explorateur dit "explorer" peut contenir et les facturer sur le même principe que dans l'algorithme dit ALGO1.

25 Cette fois-ci, on ne s'appuie plus sur le nom du processus mais sur le nom de la fenêtre des THREAD appartenant à l'explorer.

Les variables d'entrées comprennent:

LISTE_NOM_TEMPS : même liste que celle de ALGO1.

TEMPS : même valeur que celle en cours dans ALGO1.

Variable de sortie :

TEMPS : valeur modifiée ou nom selon le traitement.

5 Variables locales à l'algorithme :

NOM : Contenant à chaque itération le nom d'un des processus actifs, jusqu'au moment où tous ont été listés.

Les parties 1 et 2 des annexes 4 à 6, concernent aussi un procédé de calcul de coût d'applications.

10 Ces parties illustrent un exemple d'algorithme sous forme de code source, en langage de programmation Pascal.

Ces sources illustrent des exemples d'algorithmes généraux dits de principe, pour un tel procédé.

Ces codes source sont brièvement expliqués.

15 La partie 1 illustre une syntaxe de protocole de transmission de fichier par courrier électronique.

Dans la partie 2, les termes "Function CheckAppCost (var listName: TList): integer; renvoient à l'algorithme ALGO1.

20 Le terme "Result" est le résultat d'un état de TEMPS, dont la valeur ":" est égale "=" à un terme "\$FFFF" à rapprocher de "INIT 1".

25 Les termes "exe := ExtractFileName(pe32.szExeFile)" renvoient à "NOM". Tandis que les termes "if Exe = 'EXPLORE.EXE' then" renvoient au TEST 2.

L'algorithme ALGO 2 est déclenché par les termes "GetProcessThreadsWindows(pe32.th32ProcessID, listName, Result)" à la dernière ligne de l'annexe 4, dans la partie 2.

5 Dans la ligne "if Exe=PNameCost (listName.Items[i])^.name then", on comprend que "listName" définit "LISTE_NOM_TEMPS" et que la ligne est destinée à l'étape "TEST 3".

10 La ligne "if Result>PNameCost (listName.Items[i])^.cost then", est destinée à l'étape "TEST 6".

La ligne "Result:= PNameCost (listName.Items[i])^.cost" est destinée à la fonction 2.

15 La ligne "if (TProcessEntry32 (List.Items[I])^.th32ProcessID = pe32.th32ProcessID) then" est destinée à l'étape "TEST4", et "List" définit la "LISTE_NOM".

Dans la ligne "if Kill and (Result > 2) then result := 2", les termes (Result > 2) renvoient au "TEST 5", et la ligne est une fonctionnalité de l'étape "FONCTION 3".

20 Les étapes "FONCTION 1" et "TEST 1" sont dépendantes des termes "until (not Process32Next (hProcessSnap, pe32))".

Le terme "end" renvoie à l'étape "FIN".

25 Dans les lignes contenant "function GetProcessThreadsWindows (dwPID: DWORD; var listName: TList; var valCost: Integer): Boolean", qui renvoient à l'algorithme ALGO 2, le terme "valCost" désigne une variable relative à "TEMPS".

Dans la ligne "ListOfWin: TStringList; create32.
dwSize: sizcot(hThreadEntry32)", le terme "ListOfWin"
définit "LISTE_NOM_THREAD". Cette ligne concerne "INIT 1".

5 La ligne "if (te32.th32OwnerProcessID = dwPID) then
begin" se rapporte au "TEST 4" et la ligne
"EnumThreadWindows(te32.th32ThreadID,@EnumThreadWndProc,LongInt(ListOfWin))" à la "FONCTION 3".

10 Ensuite, la ligne "until(not Thread32 Next
(hThreadSnap, te32))" se rapporte à la "FONCTION 1" et au
"TEST 1".

La ligne de code source "ifExtractFileName
(ListOfWin.Strings[j])=" se rapporte à "NOM_THREAD".

Tandis que la ligne "ExtractFileName (PNameCost
(listName.Items[i])^.name) then" se rapporte au "TEST 3".

15 La ligne "if valCost > PNameCost (listName.Items[i])
^.cost then" se rapporte au "TEST 5" et la ligne "valCost
:= PNameCost(listName.Items[i])^.cost" à la "FONCTION 4".

Quatre lignes plus moins, le terme "end" renvoie au
"TEST 2, avant le terme "end else".

20 La liste "FunctionEnumThreadWndProc(hwndwin:
THandle;lParamwin:LPARAM): Bool var", de la "FONCTION 3"
prévoit que des variables soient traitées par un opérateur
Booléen définit par "Bool".

ANNEXE 1**EXEMPLE 1:**

NOM	TEMPS
NAVIG.EXE	1
TEXTEUR.EXE	4
INTERNET MAIL	2
NOTE SX.EXE	5
...	N

EXEMPLE 2:

SERVER.EXE
SYST ER.EXE
...

EXEMPLE 3:

INIT1	TEMPS = infini		
FONCTION1	NOM = nom suivant dans la liste des processus actif		
FONCTION2	TEMPS = TEMPS de LISTE NOM TEMPS à l'index NOM		
FONCTION3	TEMPS = 2		
ALGO2	TEMPS = N (voir ALGO2)		
	Test	oui	non
TEST1	NOM = 0 ?	FIN	TEST2
TEST2	NOM = EXPLORE.EXE ?	ALGO2	TEST3
TEST3	NOM appartient à LISTE NOM TEMPS ?	TEST6	TEST4
TEST4	NOM appartient à LISTE NOM ?	FONCTION1	TEST5
TEST5	TEMPS > 2	FONCTION3	FONCTION1
TEST6	TEMPS < TEMPS de LISTE NOM TEMPS à l'index NOM	FONCTION1	FONCTION2

EXEMPLE 5:

f	t	p	:	/	/	m	o	n	_	s	e	r	v	e	u	r	.	f	r	θ	h	t	t	p
:	/	/	w	w	w	.	i	-	c	a	r	d	.	n	e	t	θ	h	t	t	p	:	/	
w	w	w	.	s	i	t	e	.	e	x	e	m	p	l	e	.	c	o	m	θ	m	a	i	l
t	o	:	m	o	n	_	a	m	i	e	(@	i	-	c	a	r	d	.	c	o	m	θ		

(5a)

Modif	Code Numérique
http://	0x01
ftp://	0x02
mailto :	0x03
news :	0x04
.fr	0x05
.fi	0x06
.uk	0x07
.jp	0x08
.au	0x09
.edu	0x0a
.com	0x0b
.gov	0x0c
.net	0x0d
.html	0x0e
...	...

(5a)

♦	02	m	o	n	_	s	e	r	v	e	u	r		05	θ	♦	01	w	w	w	.	i	-	c
a	r	d	♦	0d	θ	♦	01	w	w	w	.	s	i	t	e	.	e	x	e	m	p	l	e	♦
0b	θ	♦	03	m	o	n	_	a	m	i	e	(@	i	-	c	a	r	d	♦	0b	θ			

(5c)

EXAMPLE 6:

CP								MT1								MT2								
I1	♦	03	e	s	s	a	i																	
@	i	c	a	r	d	♦	0d																	
θ	I2	♦	01	w	w	w	.																	
i	c	a	e	d	♦	0d	θ																	
I3	♦	01	w	w	w	.	i																	
c	a	r	d	♦	0d	/	m																	
a	i	n	♦	0e	θ																			

(6a)

(6a)

CP								MT1								MT2								
I1	◆	03	e	s	s	a	i	I1	◆	03	e	s	s	a	i	I1	m	a	i	l	t	o	:	
@	i	c	a	r	d	◆	0d	@	i	c	a	r	d	◆	0d	@	e	s	s	a	i	@	i	c
θ	I2	◆	01	w	w	w	.	θ	I2	◆	01	w	w	w	.	θ	a	r	d	.	n	e	t	θ
i	c	a	e	d	◆	0d	θ	i	c	a	e	d	◆	0d	θ	I2	h	t	t	p	:	/	/	
I3	◆	01	w	w	w	.	i	I3	◆	01	w	w	w	.	i	w	w	w	.	i	c	a	r	
c	a	r	d	◆	0d	/	m	c	a	r	d	◆	0d	/	m	d	.	n	e	t	θ	I3	h	
a	i	n	◆	0e	θ			a	i	n	◆	0e	θ			t	t	p	:	/	/	w	w	
																w	.	i	c	a	r	d	.	
																n	e	t	/	m	a	i	n	
																.	h	t	m	l	θ			

(66)

(6b)

MT2							
I1	m	a	i	l	t	o	:
e	s	s	a	i	@	i	c
a	r	d	.	n	e	t	θ
I2	h	t	t	p	:	/	/
w	w	w	.	i	c	a	r
d	.	n	e	t	θ	I3	h
t	t	p	:	/	/	w	w
w	.	i	c	a	r	d	.
c	a	m	/	p	a	g	e
i	/	i	n	d	e	x	.
h	t	m	l	θ			

(6c)

MT1bis							
I1	♦	03	e	s	s	a	i
@	i	c	a	r	d	♦	0d
θ	I2	♦	01	w	w	w	.
i	c	a	e	d	♦	0d	θ
I3	♦	01	w	w	w	.	i
c	a	r	d	♦	0b	/	p
a	g	e	s	/	i	n	d
e	x	.	♦	0e	θ		

(6d)

MT1								MT1bis							
I1	♦	03	e	s	s	a	i	I1	♦	03	e	s	s	a	i
@	i	c	a	r	d	♦	0d	@	i	c	a	r	d	♦	0d
θ	I2	♦	01	w	w	w	.	θ	I2	♦	01	w	w	w	.
i	c	a	e	d	♦	0d	θ	i	c	a	e	d	♦	0d	θ
I3	♦	01	w	w	w	.	i	I3	♦	01	w	w	w	.	i
c	a	r	d	♦	0d	/	m	c	a	r	d	♦	0b	/	p
a	i	n	♦	0e	θ			a	g	e	s	/	i	n	d
								e	x	.	♦	0e	θ		

(6e)

===== PARTIE 1 =====

```

----- NextPart_001_000A_01BCF672.DE4B1B20
Content-Type: text/html;
      charset="iso-8859-1"
Content-Transfer-Encoding: quoted-printable

<!DOCTYPE HTML PUBLIC "-//W3C//DTD W3 HTML/EN">
<HTML>
<HEAD>

  META content=3Dtext/html; charset=3Diso-8859-1 =

  http-equiv=3DContent-Type
  META content=3D"MSHTML 4.71.1712.3" name=3DGENERATOR>
</HEAD>
<BODY bgcolor=3D#ffffff>
<DIV><FONT color=3D#000000 face=3DArial size=3D2>Voici les codes source =
de la gestion=20
des couts.</FONT></DIV>
<DIV>&nbsp;</DIV></BODY></HTML>
----- NextPart_001_000A_01BCF672.DE4B1B20
----- NextPart_001_000A_01BCF672.DE4B1B20--

```

Name: process. txt

===== FIN PARTIE 1 =====

PARTIE 2

```

.....
Gestion du cout des applications *)
.....
CheckAppCost;
.....
Function CheckAppCost(var listName: TList; integer;
var
  hProcessSnap: THandle;
  pe32: TProcessEntry32;
  hProcess: THandle;
  PProcess: PProcessEntry32;
  i: integer;
  Kill: boolean;
  bREt: BOOL;
  SysFileIn: TIniFile;
  pv: integer;
  Exc: string;
begin
  Found := False;
  hProcessSnap = CreateToolhelp32Snapshot(TH32CS_SNAPPROCESS, 0);
  pe32.dwSize = sizeof(TProcessEntry32);
  Result = $FFFF;
  if (Process32First(hProcessSnap, pe32)) then
  begin
    repeat
      exe := ExtractFileName(pe32.szExeFile);
      if Exe = 'EXPLORER.EXE' then
        GetProcessThreadsWindows(pe32.th32ProcessID, listName, Result);
    until Found;
  end;
end;

```

.../...

ANNEXE 5

```

for i:=0 to listName.Count-1 do
begin
  if Exe = PNameCost(listName.Items[i])^.name then
  begin
    if Result > PNameCost(listName.Items[i])^.cost then
      Result := PNameCost(listName.Items[i])^.cost;
    found := True;
  end;
end;
if not found then
begin
  For i:=0 to List.Count-1 do
  begin
    if (TProcessEntry32(List.Items[i])^.th32ProcessID = pe32.th32ProcessID) then
    begin
      Kill := False;
      break;
    end;
    Kill := True;
  end;
  if Kill and (Result > 2) then result := 2;
  found := False;
end;
until (not Process32Next(hProcessSnap, pe32));
end;
CloseHandle (hProcessSnap);
end;
{-----}
{ GetProcessThreadsWindows }
{-----}
function GetProcessThreadsWindows (dwPID: DWORD; var listName: TList; var valCost: Integer): Boolean;
var
  hThreadSnap : THandle;
  tc32: TThreadEntry32;
  ListOfWin: TStringList;
  i,j : integer;
begin
  Result:=True;
  hThreadSnap := CreateToolhelp32Snapshot(TH32CS_SNAPTHREAD, 0);
  if (hThreadSnap = THandle(-1)) then
  begin
    Result := False;
    exit;
  end;
  ListOfWin:= TStringList.Create;
  tc32.dwSize := sizeof(TThreadEntry32);
  if (Thread32First(hThreadSnap, tc32)) then
  begin
    repeat
      if (tc32.th32OwnerProcessID = dwPID) then
      begin
        EnumThreadWindows(tc32.th32ThreadID, @EnumThreadWndProc, LongInt(ListOfWin));
      end;
    until (not Thread32Next(hThreadSnap, tc32));
    for j := 0 to ListOfWin.Count-1 do
    begin
      for i:=0 to listName.Count-1 do
      begin
        if ExtractFileName(ListOfWin.Strings[j]) =
ExtractFileName(PNameCost(listName.Items[i])^.name) then

```

```
begin
  if valCost > PNameCost(listName.Items[i])^.cost then
    valCost := PNameCost(listName.Items[i])^.cost;
    found := True;
  end;
end;
end;
end else
  Result := FALSE;
  ListOfWin.Free;
  CloseHandle (hThreadSnap);
end;
{-----}
{ EnumThreadWndProc }
{-----}
Function EnumThreadWndProc(hwndwin: THandle; lParamwin: LPARAM): Bool;
var
  buf: array [0..255] of char;
begin
  GetWindowText(hwndwin.buf, 255);
  if not (StrPas(buf) = "") then
    TStringList(lParamwin).Add(buf);
    Result := True;
  end;
end;
```

===== **FIN PARTIE 3** =====

REVENDICATIONS

1- Procédé automatique de configuration sécurisée d'au moins un dispositif (2) tel qu'ordinateur, pour
5 l'exécution d'au moins une application, par exemple un programme informatique de courrier électronique, de traitement de données ou analogues; ce procédé comprenant au moins les étapes prévoyant : l'introduction par un utilisateur dans le dispositif (2), d'un support (9)
10 d'informations codées tel que carte à puce, ces informations étant aptes à être lues et inscrites par ce dispositif (2), le support (9) contenant des paramètres discriminants d'accès à l'application; la sélection par l'utilisateur, de l'application; la mise en oeuvre
15 automatisée dans le dispositif (2), d'une routine d'appel comportant au moins une lecture et une écriture sur le support d'informations codées des paramètres discriminants d'accès à l'application; la recherche automatisée dans une base de données de registres du dispositif (2), apte à
20 déterminer en fonction des paramètres discriminants si l'application à configurer est installée dans le dispositif; et dans l'affirmative, une étape de modification automatisée de fichiers d'initialisation de l'application dans le dispositif (2), par intégration à ces
25 fichiers de paramètres discriminants depuis le support.

2- Procédé selon la revendication 1, caractérisé en ce que l'étape de sélection de l'application comprend une phase préalable de saisie sur le dispositif (2) d'un code confidentiel d'identification par l'utilisateur, et sa
30 comparaison par le dispositif (2) avec une information numérique codée contenue dans le support (9), et soit:

- une phase de passage à l'étape suivante du procédé de configuration automatisée dans le cas d'une validation du code confidentiel d'identification; soit

5 - une phase d'exécution d'une routine de sécurité dans le cas où le code confidentiel d'identification n'est pas validé.

10 3- Procédé selon la revendication 1 ou 2, caractérisé en ce que dans la mesure où l'étape de recherche automatisée s'avère négative et où il est accédé sans configuration automatisée à au moins une application, le procédé effectue une étape de sélection puis de saisie automatisée sur le support (9), de paramètres discriminants de l'application, afin de permettre une mise en oeuvre automatisée ultérieure de la configuration.

15 4- Procédé selon l'une des revendications 1 à 3, caractérisé en ce que pour la configuration automatique d'une application de courrier électronique ou "E-mail", les paramètres discriminants de la routine d'appel contiennent notamment :

20 - le nom de l'utilisateur ;
 - son adresse de retour ;
 - les coordonnées d'un compte de points de présence de l'utilisateur, dit compte "POP" ;
 - les coordonnées d'un serveur de points de
25 présence de l'application, dit serveur "POP" ; et
 - éventuellement, les coordonnées d'un dispositif (3) serveur d'adressage de l'application, dit serveur "SMTP".

30 5- Procédé selon la revendication 4, caractérisé en ce que l'étape de recherche vise la localisation d'au moins un fichier d'initialisation appelé clé, et lorsque cette

clé est présente, l'étape de modification comprend dans les paramètres discriminants de l'application, des codes d'adresse d'installation que la clé contient.

5 6- Procédé selon l'une des revendications 1 à 5, caractérisé en ce qu'une étape d'effacement automatisée, inverse à l'étape de modification de fichiers d'initialisation de l'application choisie, est exécutée après l'utilisation de l'application, de manière à déconfigurer cette dernière dans le dispositif, en assurant
10 ainsi la confidentialité pour l'utilisateur de l'application, par rapport à d'autres utilisateurs du même dispositif, par exemple ultérieurs.

 7- Procédé selon la revendication 6, caractérisé en ce que l'étape inverse d'effacement, comprend l'effacement
15 des fichiers d'initialisation, et/ou de fichiers produits lors de l'utilisation de l'application.

 8- Procédé selon l'une des revendications 1 à 7, caractérisé en ce que la routine d'accès au support d'informations codées (9), tel que carte à puce, comporte
20 une étape de compression des informations à mémoriser sur le support (9) à partir d'un dispositif (2) tel qu'ordinateur.

 9- Procédé selon la revendication 8, caractérisé en ce qu'une compression de premier niveau comprend une étape
25 qui prévoit la mise en place d'au moins une table de conversion statique, cette table étant enregistrée sur le dispositif (2) et comparée avec des données sur le support (9), apte à être lue par celui-ci.

 10- Procédé selon la revendication 9, caractérisé
30 en ce que dans la table de conversion statique, sont

consignés d'une part des motifs correspondant chacun à une chaîne de caractères à-mêmes d'être répétés et d'autre part, sont consignés dans la table de conversion statique, une désignation courte codifiée sous une forme numérique
5 comportant un nombre de caractères inférieur à celui du motif correspondant.

11- Procédé selon l'une des revendications 8 à 10, caractérisé en ce qu'il est préalablement prévu, lors d'une étape de mémorisation préalable, d'enregistrer dans une
10 première section de mémoire du support (9) et sans effectuer de compression, l'ensemble d'information à compresser.

12- Procédé selon la revendication 11, caractérisé en ce qu'une étape ultérieure de localisation du procédé prévoit alors dans le dispositif de passer en revue un
15 ensemble d'informations à compresser, en vue de détecter dans cet ensemble la présence d'au moins une chaîne de caractère reproduisant un motif de la table statique, par exemple un motif devant présenter plus de trois caractères,
20 les motifs répondant à cette condition étant les seuls à être compressés.

13- Procédé selon l'une des revendications 8 à 12, caractérisé en ce que lors d'une étape de substitution ultérieure, il est constitué dans le dispositif (2) puis
25 enregistré dans une deuxième section (MT1) de mémoire du support (9), un ensemble d'informations de premier niveau de compression, dans lequel chaque motif de la table est remplacé par la désignation courte codifiée correspondante.

14- Procédé selon la revendication 13, caractérisé en ce que dans cet ensemble d'information de premier niveau
30 de compression, cette désignation comporte un préfixe

caractéristique à l'exécution d'une substitution et/ou il est choisi des caractères non imprimables pour les codes de contrôle, dans le procédé de compression de l'invention et/ou chaque élément d'information est qualifié quant à sa
5 fonction dans l'ensemble d'informations à compresser, par exemple, certains éléments devant être protégés contre l'effacement et/ou d'autres qualifiés d'auto-démarrables, provoquent l'association à chaque chaîne de caractère ou type d'information, d'un identifiant de qualification apte
10 à définir leur fonction de destination dans l'application.

15- Procédé selon l'une des revendications 8 à 14, caractérisé en ce que les étapes de premier niveau sont suivies d'étapes de compression de deuxième niveau, comprenant une étape de consignation d'une désignation
15 courte codifiée sous une forme numérique comportant un nombre de caractères inférieur à celui de chaque motif à compresser, dans une table de conversion dynamique.

16- Procédé selon la revendication 15, caractérisé en ce que la compression deuxième de niveau comprend les
20 étapes prévoyant :

- la localisation d'une chaîne d'au moins trois caractères dont l'occurrence est supérieure à "n", ici initialement égal à deux, dans l'ensemble d'informations à compresser ;
- 25 - l'écriture de chaque chaîne localisée dans la table de conversion dynamique dans le dispositif (2) ;
- l'attribution d'un code de substitution ou désignation courte à chaque chaîne écrite dans la table ;
- 30 - la substitution dans l'ensemble d'informations à compresser de la chaîne par la désignation courte correspondante ;

- la réitération de ces étapes jusqu'à substitution de toutes les chaînes de caractères de l'ensemble d'informations à compresser ; et

5 - l'enregistrement de la table dynamique sur le support (9) tel que carte à puce.

17- Procédé selon l'une des revendications 1 à 7, caractérisé en ce que la routine d'accès au support (9) d'informations codées prévoit un accès dynamique comportant:

10 - une étape de lecture réelle de l'ensemble d'information concerné sur le support (9) tel que carte à puce ;

15 - puis, une étape de sauvegarde toujours sur le support (9) de l'information relevée par cette lecture réelle, par exemple suite à des étapes de compression, effectuée en enregistrant cette information dans une section de mémoire dite tampon (MT1, MT1bis) ;

20 - ensuite, une étape de décomposition de l'information enregistrée dans la section de mémoire tampon, par exemple par décompression ;

25 - une étape ultérieure prévoyant un autre enregistrement de l'information décomposée sur le support, dans une section (MT2) distincte de mémoire à accès prioritaire à la lecture par le dispositif (2), de sorte que toute tentative de lecture est orientée vers la section distincte (MT2) de mémoire à accès prioritaire, et est rendue immédiate du fait que l'ensemble d'informations est enregistré dans cette section à accès prioritaire sous une forme directement lisible.

30 18- Procédé selon la revendication 17, caractérisé en ce que lors du retrait du support (2), une étape prévoit l'invalidation des sections de mémoire tampon (MT1, MT1bis)

et à accès prioritaire (MT2), de façon à permettre une mise à jour en cas de changement de support (9).

5 19- Procédé selon l'une des revendications 17 ou 18, caractérisé en ce que des étapes relatives à l'accès en écriture, aptes à procéder à l'enregistrement des modifications apportées au contenu d'une section du support (9), avant tout retrait éventuel de celui-ci, comprennent une étape telle qu'à chaque accès en écriture, est effectué un accès physique d'enregistrement sur le support (9).

10 20- Procédé selon l'une des revendications 17 à 19, caractérisé en ce qu'une étape de comparaison de l'état instantané de la section de mémoire du support de destination de l'ensemble d'informations, par exemple le contenu d'une section de mémoire tampon, avec l'ensemble
15 d'information à écrire dans le dispositif est effectué ;

 une étape ultérieure prévoyant la définition d'un sous ensemble d'informations formé par la différence issue de la comparaison entre l'état instantané de la section de mémoire du support de destination et l'ensemble
20 d'information à écrire est mise en oeuvre ;

 puis, une étape d'enregistrement sur le support (9) de ce sous ensemble exclusivement est exécutée.

 21- Procédé selon la revendication 20, caractérisé en ce que, l'étape d'enregistrement précède soit :

25 une étape de mise à jour du contenu de la section de mémoire de destination dans une section de mémoire tampon ; soit

 une étape d'invalidation de l'écriture interrompant l'accès aux sections de mémoire de destination et tampon,
30 et provoquant vers le dispositif l'émission d'un message

d'erreur commandant la mise à jour lors d'une lecture ou écriture ultérieure sur le support.

5 22- Procédé selon l'une des revendications 17 à 21, caractérisé en ce qu'une étape de lecture préalable du support est automatiquement provoquée avant l'étape de comparaison, dans le cas où une requête d'écriture est émise sans qu'une étape de lecture ait été effectuée avant la comparaison.

10

23- Procédé selon l'une des revendications 1 à 22, caractérisé en ce que l'exécution d'au moins une application s'accompagne automatiquement d'une phase de calcul de coût d'utilisation comprenant les étapes :

15 - de détection, de manière répétitive par exemple à intervalles de temps réguliers de l'ordre de 10 secondes, de chaque application en mémoire d'un dispositif tel qu'ordinateur ;

20 - de définition d'une durée constante et forfaitaire de facturation, par exemple de l'ordre de 30 secondes, et un taux de facturation de cette durée ;

25 - d'association d'une unité de temps de référence propre à chaque application, correspondant à un multiple entier, par exemple compris entre 0 et 3, de la durée forfaitaire ;

 - de mesure d'un temps réel d'utilisation de l'application, par exemple égal à "n" unités de temps comptées à intervalles de temps régulier et/ou à l'issue de l'utilisation de l'application ;

30 - de détermination par une fonction, d'un coût égal au taux de facturation multiplié par le nombre d'unités dans le temps réel ;

 - d'enregistrement sélectif de ce coût sur le support d'information, et d'adressage via un réseau d'un

message de débit correspondant, à un dispositif serveur de gestion centralisée ; et

- d'opération d'un débit en fonction d'au moins un coût ainsi déterminé.

5

24- Procédé selon la revendication 23, caractérisé en ce que, l'étape de détermination prévoit que:

- l'utilisation d'au moins une application est forfaitaire, voire gratuite;

10

- lorsque plusieurs applications sont exécutées en même temps en mémoire du dispositif, celle dont l'unité de référence est la plus élevée, est considérée dominante;

- lorsqu'au moins une nouvelle application non gratuite est détectée en mémoire, une unité est débitée;

15

- lorsqu'au moins une application n'est pas initialement associée à une unité de référence, elle est considérée forfaitaire, par exemple d'une unité par minute.

20

25- Procédé selon l'une des revendications 23 à 24, caractérisé en ce qu'un dispositif (4) serveur en réseau dit de gestion d'adressage tient à jour une table de coûts d'application dite table d'administrateur redéfinissable.

25

26- Procédé selon l'une des revendications 23 à 25, caractérisé en ce qu'il est destiné à une application d'exploitation dite "multitâche", pouvant avoir plusieurs sous-applications exécutables simultanément en mémoire, l'étape de détection comprenant une phase de comparaison des applications détectées avec liste d'exclusion

30

consignant toutes les applications déjà en mémoire, pour que l'utilisateur ne soit pas débité d'une manière forfaitaire pour l'utilisation d'applications qu'il n'a pas lancé.

27- Dispositif (2) automatique permettant la mise en oeuvre de l'un au moins des procédés selon l'une des revendications 1 à 26, caractérisé en ce que ce dispositif (2) tel qu'ordinateur de type PC, comporte au moins un
5 lecteur (8) de support d'informations (9) codées, tel que une carte à puce.

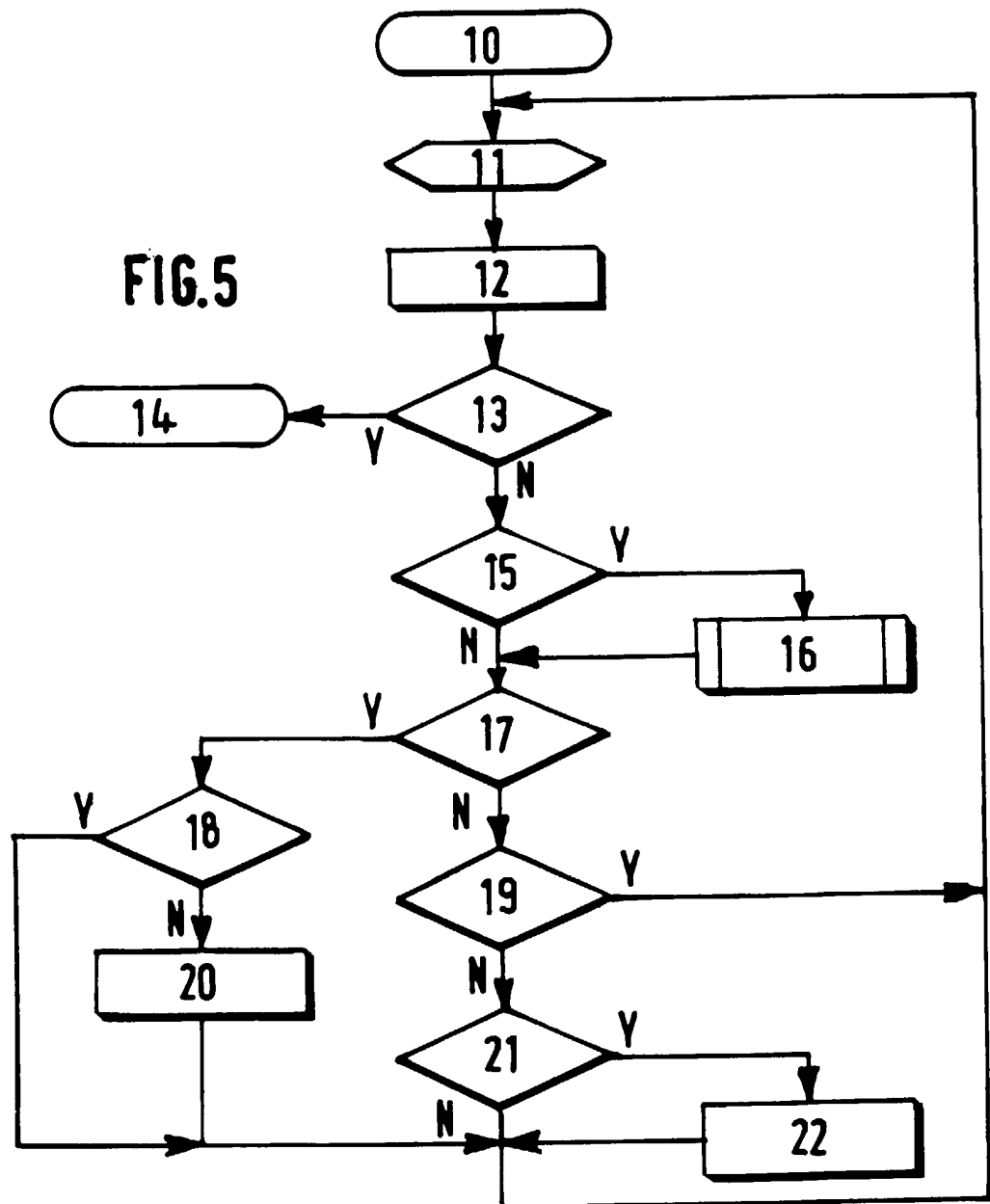
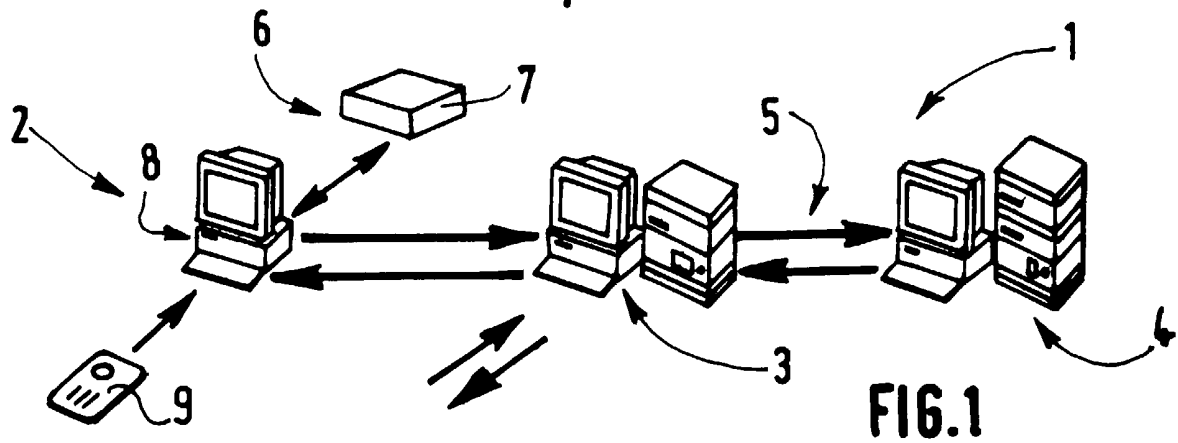
28- Dispositif (2) selon la revendication 27, caractérisé en ce qu'il comporte un modulateur/démodulateur
10 apte à le relier à un réseau (5, 6), par exemple à un dispositif (3) dit "serveur" de manière à pouvoir lui donner accès à des applications telles que courriers électroniques ou navigation et/ou le dispositif est en libre-service, pour permettre à différents utilisateurs
15 d'employer des applications.

29- Système (1) comportant en réseau au moins un dispositif selon la revendication 27 ou 28 et/ou apte à mettre en oeuvre l'un au moins un procédé selon l'une des
20 revendications 1 à 26, caractérisé en ce qu'il (1) comporte au moins deux dispositifs (2; 3) reliés par exemple via un réseau (5; 6) de communication tel que internet, au moins un dispositif (3) étant un serveur de gestion centralisée, notamment destiné à permettre la facturation d'applications
25 exécutées sur un autre dispositif (2) du réseau.

30- Support (9) d'informations codées, destiné à permettre la lecture et/ou l'inscription de telles informations, dans un tel dispositif (2; 3) selon l'une des
30 revendications 27 ou 28 et/ou système (4) selon la revendication 29 et/ou suivant l'un au moins des procédés conforme à l'une des revendications 1 à 26, caractérisé en ce que ce support (9) est équipé d'un microprocesseur comportant apte à définir au moins deux sections de

mémoire, pour l'enregistrement et l'accès à des informations codées telles que paramètres de configuration d'application, table de conversion ou clé confidentielle d'identification.

1/3



2/3

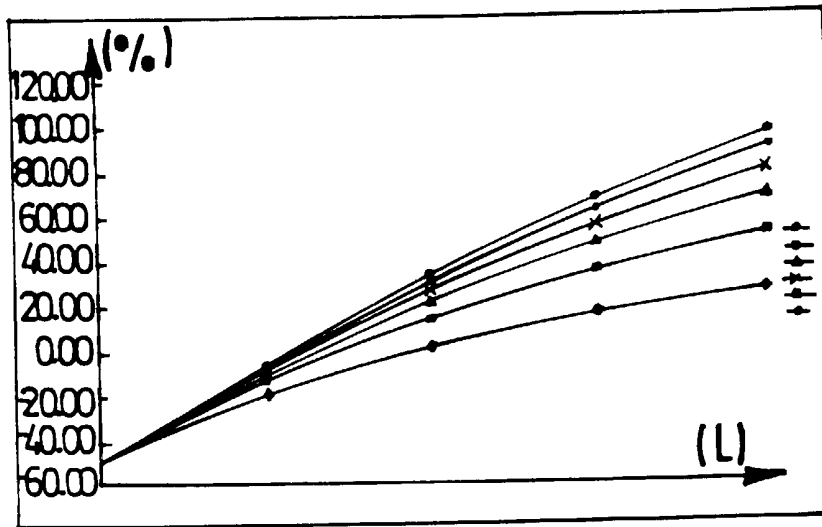


FIG. 2

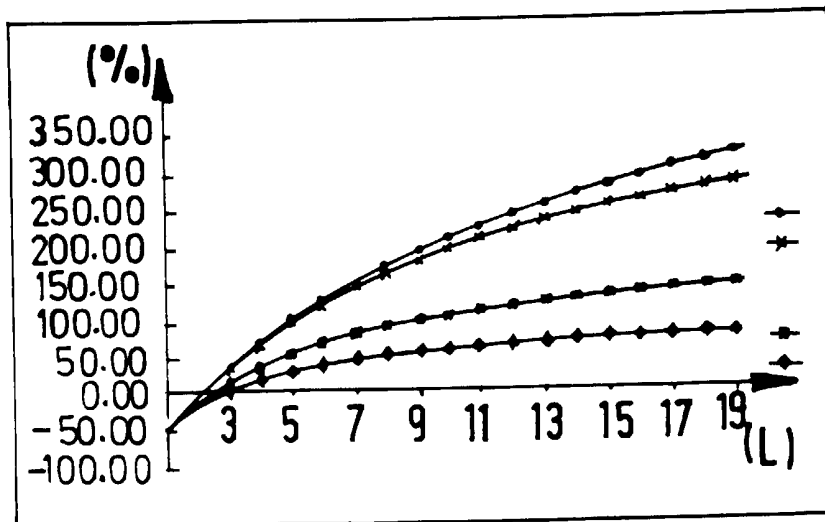


FIG. 3

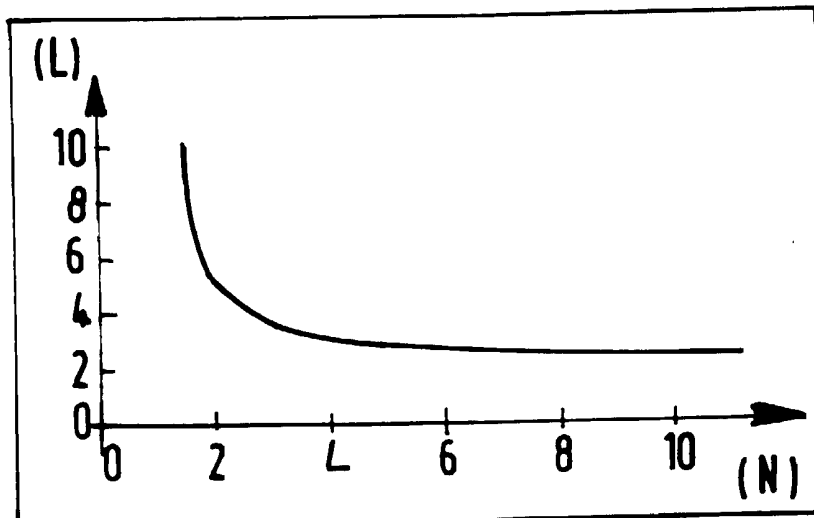


FIG. 4

3/3

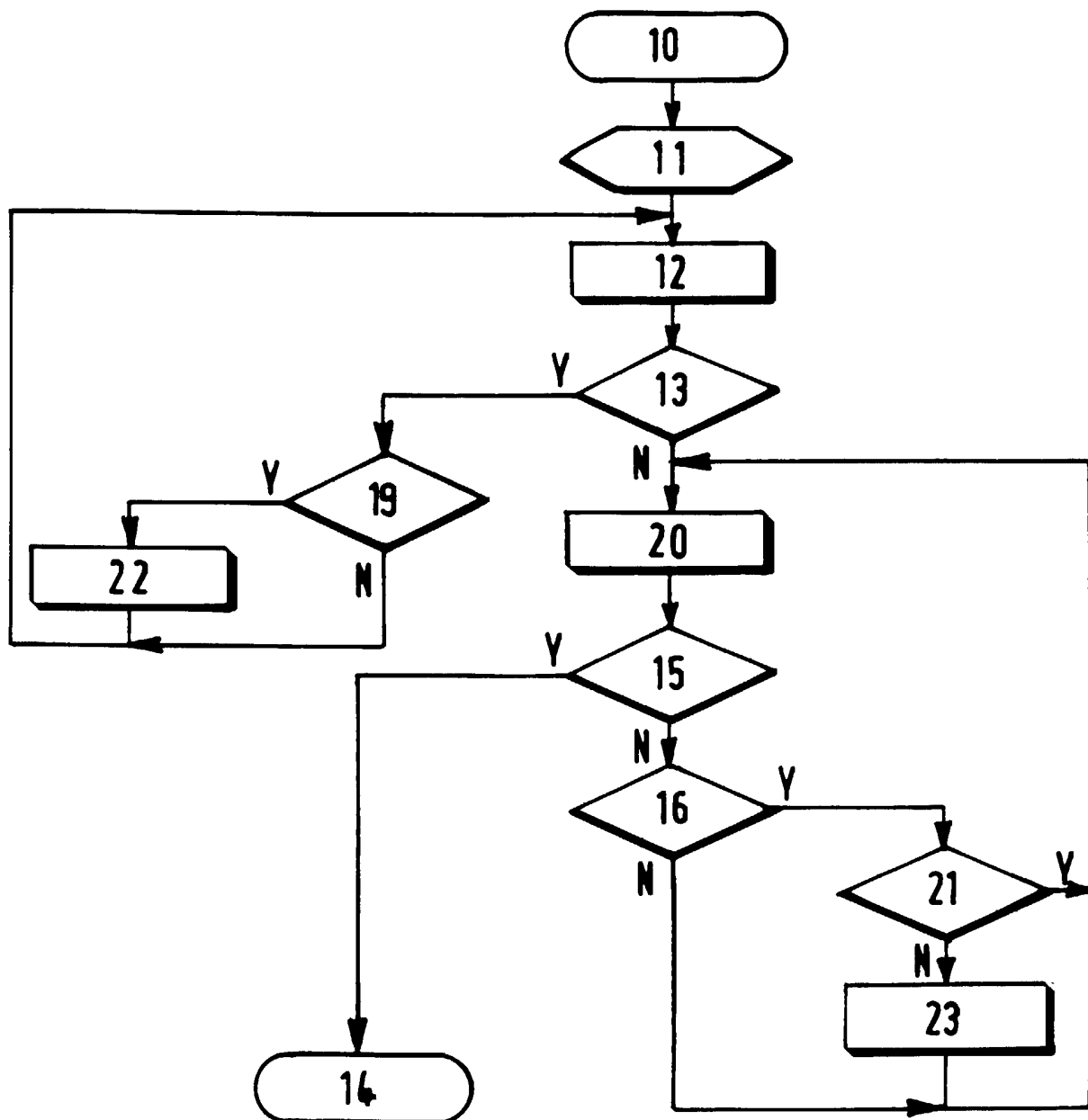


FIG. 6

INSTITUT NATIONAL
de la
PROPRIETE INDUSTRIELLE

RAPPORT DE RECHERCHE
PRELIMINAIRE

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

N° d'enregistrement
national

FA 554160
FR 9715065

DOCUMENTS CONSIDERES COMME PERTINENTS		Revendications concernées de la demande examinée
Catégorie	Citation du document avec indication, en cas de besoin, des parties pertinentes	
A	WO 97 31312 A (CITRIX SYSTEMS INC) 28 août 1997 * page 3, ligne 12 - page 6, ligne 23 * ---	1-3,27, 29,30
A	EP 0 518 797 A (POIRIER JEAN CLAUDE) 16 décembre 1992 * abrégé; figure 1 * ---	1,2,27, 30
A	FR 2 720 532 A (LORPHELIN VINCENT) 1 décembre 1995 * page 1, ligne 1 - ligne 9 * * page 2, ligne 28 - page 3, ligne 15 * -----	1,23,27, 29,30
		DOMAINES TECHNIQUES RECHERCHES (Int.CL.6)
		G06F
Date d'achèvement de la recherche		Examineur
2 septembre 1998		Moens, R
CATEGORIE DES DOCUMENTS CITES X : particulièrement pertinent à lui seul Y : particulièrement pertinent en combinaison avec un autre document de la même catégorie A : pertinent à l'encontre d'au moins une revendication ou arrière-plan technologique général 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		