

Señores
SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO
DIVISIÓN DE NUEVAS CREACIONES
E. S. D.

79

28

SUPERINDUSTRIA Y COMERCIO
Radicación: 04094972 00000002 Folios: 92
Fecha (HH): 2004-12-09 10:20:55
Trámite: 002 PATENTES 1 REGISTRO/ 444 RESPUESTA
Dependencia: 2020 DIVISION DE NUEVAS CREACIONES

Ref.: Expediente No. 04-094.972
Solicitud de Patente para "INTERFASE DE PROGRAMACIÓN
PARA EL LICENCIAMIENTO".

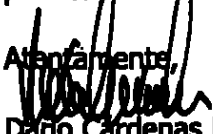
Solicitante: Microsoft Corporation.

Darío Cárdenas Navas, identificado como aparece al pie de mi firma, atentamente me dirijo a ese Despacho dando respuesta al Oficio No. 10015 de fecha 28 de Octubre de 2004.

Para tal efecto me permito adjuntar los siguientes documentos:

1. Copia auténtica y debidamente Apostillada del documento donde consta la cesión del derecho a la patente otorgada por los inventores a favor del solicitante.
2. Copia auténtica del documento de prioridad correspondiente a la patente de los Estados Unidos No. 10.692.868 de fecha 24 de Octubre de 2003. Tal como consta en los archivos de la Oficina de Patentes, el recibo oficial 04-075.290 de fecha 22 de Septiembre de 2004 correspondiente a la reivindicación de la prioridad fue presentado junto con la solicitud de patente radicada ante ese Despacho el día 23 de Septiembre de 2004.

Habiendo dado cumplimiento a los requerimientos incluidos en el Oficio No. 10015 de fecha 28 de Octubre de 2004 atentamente solicito a ese Despacho proceda con la publicación de la solicitud de patente en referencia.

Atentamente,

Darío Cárdenas Navas.
c.c. 17.066.629 de Bogotá
t.p. 1.250 C.S.J. 

UNITED STATES OF AMERICA

80

The State of Washington



Secretary of State

APOSTILLE

(Convention de la Haye du 5 Octobre 1961)

1. Country: **United States of America**

2. This public document has been signed by:

LAURIE A. HENEGHAN

3. acting in the capacity of:

Notary Public, state of Washington

4. bears the seal/stamp of:

LAURIE A. HENEGHAN

CERTIFIED

5. at: **Olympia, Washington**

6. the: 31 day of August, 2004

7. by: **Sam Reed, Secretary of State**

8. No: 200422875

9. Seal/Stamp:

10. Signature:



Given under my hand and the Seal of the State of Washington at Olympia, the State Capital

Sam Reed, Secretary of State



81

Colombia
305442.23

ASSIGNMENT

Between the undersigned, Caglar Gunyakti, Ning Zhang, and Wen-Pin Scott Hsu, citizens of Turkey, U.S.A., and Taiwan (R.O.C.), respectively, domiciled at One Microsoft Way, Redmond, WA 98052-6399, USA; One Microsoft Way, Redmond, WA 98052-6399, USA; and One Microsoft Way, Redmond, WA 98052-6399, USA, respectively, henceforth the Assignors, and David Bartley Eppenauer, acting in his capacity of Assistant Corporate Secretary, of Microsoft Corporation, a company legally constituted and in existence in accordance to the laws of State of Washington, USA, and domiciled at One Microsoft Way, Redmond, WA 98052-6399, USA, henceforth the Assignee and,

A. Whereas the Assignee has applied for registration before the Colombian Patent Office, the following patent application:

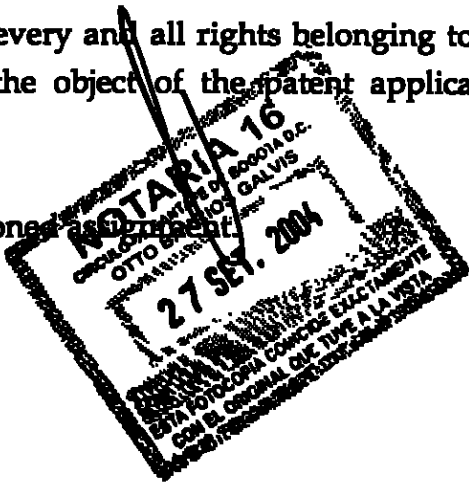
Title of the invention	File No.	Filing date
PROGRAMMING INTERFACE FOR LICENSING		

- B. Whereas the Assignors are the inventors of the invention which is the object of the above-mentioned patent application.
- C. Whereas the Assignee is interested in obtaining the assignment of the above-mentioned patent invention, and the Assignors are interested in assigning it to the Assignee.

The following agreement has been entered:

First: The Assignors assign to the Assignee every and all rights belonging to the Assignors regarding the invention that is the object of the patent application mentioned in letter A. of this document.

Second: The Assignee accepts the aforementioned assignment.

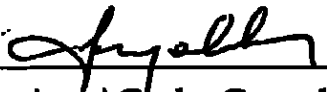


3

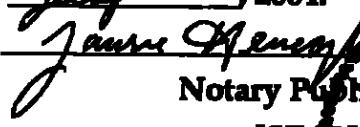
23
82

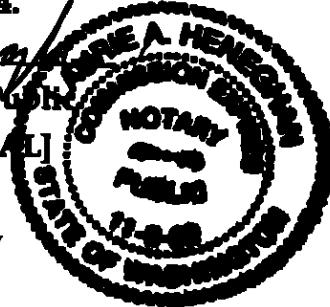
Third: This assignment shall inure to and be binding on the parties, and their subsidiaries, affiliates, successors, assignees, and licenses, if any.

Signed on the 15th day of July of 2004, in two originals, by


(Assignor) Caglar Gunyakti

Subscribed and sworn to before me, Lauree Heneghan
a Notary Public, this 15 day of July, 2004.

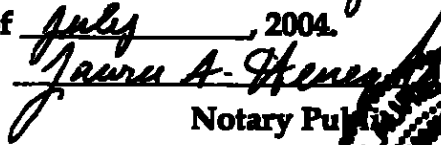

Notary Public
[SEAL]

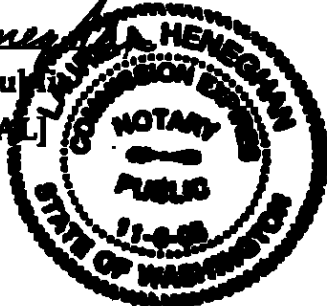


Signed on the 15th day of July of 2004, in two originals, by

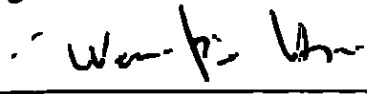

(Assignor) Ning Zhang

Subscribed and sworn to before me, Lauree Heneghan
a Notary Public, this 15 day of July, 2004.

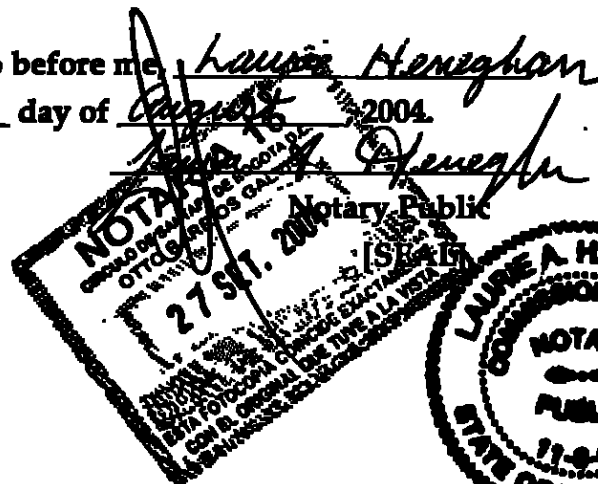

Notary Public
[SEAL]



Signed on the 17 day of Aug. of 2004, in two originals, by


(Assignor) Wen-Pin Scott Hsu

Subscribed and sworn to before me, Lauree Heneghan
a Notary Public, this 17 day of August, 2004.



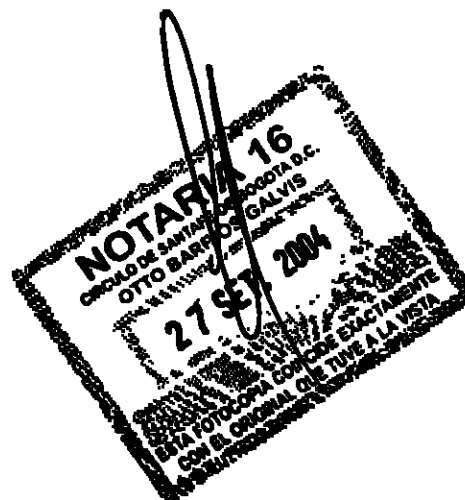
Signed on the 23rd day of August of 2004, in two originals, by

MICROSOFT CORPORATION

By: *[Signature]*
(Assignee)
David Bartley Eppenauer
Assistant Corporate Secretary

Subscribed and sworn to before me, Laurie Heneghan,
a Notary Public, this 23 day of August, 2004.
[Signature]
Notary Public

(This document must be duly legalized by Apostille)



[Handwritten mark]

Estados Unidos de América

Estado de Washington

(Sello del Estado de Washington)

Secretaría del Estado

APOSTILLE

(Convención de La Haya del 5 de octubre de 1961)

1. País: **Estados Unidos de América**
2. Este documento público ha sido firmado por **LAURIE A. HENEGHAN**
3. actuando en calidad de **Notario Público, estado de Washington**
4. portando el sello y/o estampilla de **LAURIE A. HENEGHAN**

CERTIFICADO

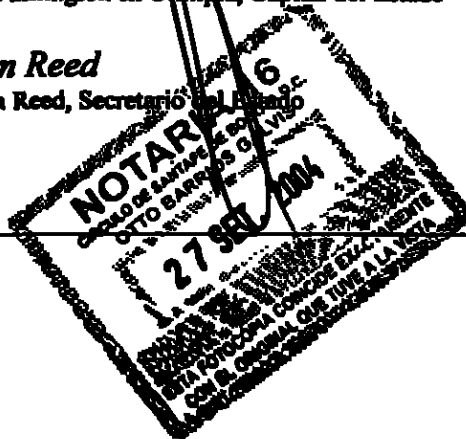
5. en **Olimpia, Washington**
6. el día **31 de agosto de 2004**
7. por **Sam Reed, Secretario del Estado**
8. No. **200422875**
9. Sello y/o Estampilla
10. Firma

**Gran Sello
en Relieve del
Estado de
Washington**

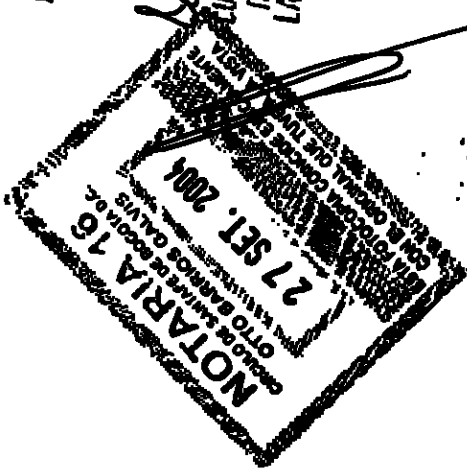
Dado mi puño y letra y con el Sello del Estado
de Washington en Olympia, Capital del Estado

Sam Reed

Sam Reed, Secretario del Estado



Victoria Eugenia Cruz de Ramirez
TRADUCTOR E INTERPRETE OFICIAL
REG. 361 - 22/04/02
MANUELITA



MINISTERIO DE RELACIONES
EXTERIORES

447692
Verónica de la Cruz
CUYA FIRMA APARECE EN EL
DOCUMENTO QUE SE PRESENTA
LAS FUNCIONES INDICADAS

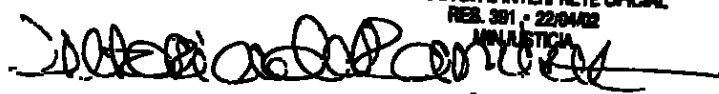
NO SE ASUME
RESPONSABILIDAD DEL TEXTO
2004 SEP 24 P 12:05
ms
OFICE DE LEGALIZACIONES
SANTAFE DE BOGOTA D.C.

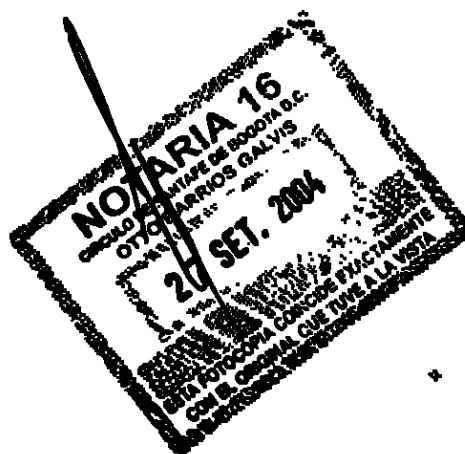
CERTIFICACIÓN

Yo, VICTORIA EUGENIA CRUZ DE RAMÍREZ, declaro que soy Traductora e Intérprete Oficial los idiomas Inglés y Español, con sede en la Calle 148 # 14-09 (202) en Bogotá DC, Colombia.

Declaro que la traducción adjunta es una versión fiel y completa al español de la APOSTILLA, expedida en Olimpia, Washington, con fecha 31 de agosto de 2004, en idioma inglés, distinguida con el número 200422875 y firmada por SAM REED, Secretario de Estado del Estado de Washington, Estados Unidos de América.

Victoria Eugenia Cruz de Ramirez
TRADUCTOR E INTERPRETE OFICIAL
RES. 391 - 22/04/02
MINJUSTICIA


VICTORIA EUGENIA CRUZ DE RAMÍREZ
Traductor e Intérprete Oficial
Resol. 0391-22/04/02 de MinJusticia



87
86

Traducción:

Colombia
305442.23

CESION

Entre los abajo firmantes, Caglar Gunyakti, Ning Zhang, y Wen-Pin Scott Hsu, ciudadanos de Turquía, EEUU y Taiwán (R.O.C.), respectivamente, con domicilio en One Microsoft Way, Redmond, WA 98052-6399, USA; One Microsoft Way, Redmond, WA 98052-6399, USA y One Microsoft Way, Redmond, WA 98052-6399, USA, respectivamente, en adelante los Cedentes, y David Bartley Eppenauer, en su calidad de Vicesecretario Corporativo de Microsoft Corporation, sociedad constituida legalmente y con existencia conforme a las leyes del Estado de Washington, Estados Unidos de América, y con domicilio en One Microsoft Way, Redmond, WA 98052-6399, USA, en adelante la Ceslonaria, y

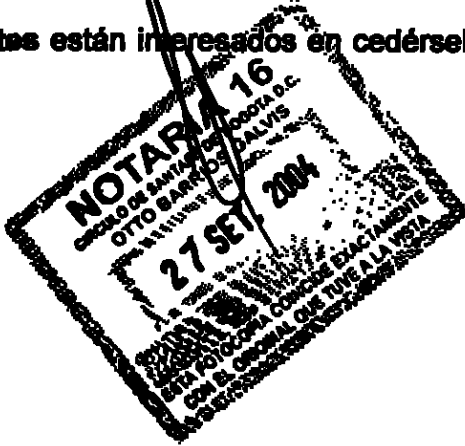
A. Considerando que la Ceslonaria ha presentado ante la Oficina Colombiana de Marcas y Patentes la siguiente solicitud de patente:

NOMBRE DEL INVENTO	EXPEDIENTE No.	FECHA DEL EXPEDIENTE
INTERFACE DE PROGRAMACIÓN PARA EL LICENCIAMIENTO		

B. Considerando que los Cedentes son los inventores del invento objeto de la solicitud de patente anteriormente mencionada.

C. Considerando que la Ceslonaria está interesada en obtener el invento con patente anteriormente citado, y los Cedentes están interesados en cedérselo a la Ceslonaria,

Se ha establecido el siguiente contrato:



Victoria Eugenia Cruz de Ruiz
TRADUCTOR E INTERPRETE OFICIAL
REG. 391 • 22/04/02
MINISTERIO DE JUSTICIA

8

88
87

Primero: Los Cedentes le ceden a la Cesionaria todos y cada uno de los derechos pertenecientes a la Cedente con respecto al invento objeto de la solicitud de patente mencionada en el numeral A del presente documento.

Segundo: La Cesionaria acepta la cesión anteriormente citada.

Tercero: Esta Cesión entrará en vigor y obligará legalmente a las partes, a sus subsidiarias, afiliados, sucesores, cesionarios y licencias, si los hubiera.

Firmado el día 15 de julio de 2004, en dos originales, por

(Firma)

(Cedente) Caglar Gunyakti

Firmada y Juramentada ante mí, Laurie Heneghar

Notario Público, hoy día 15 de Julio de 2004

Laurie Heneghar

Notario Público

SELLO

Firmado el día 15 de julio de 2004, en dos originales, por

(Firma)

(Cedente) Ning Zhang

Firmada y Juramentada ante mí, Laurie Heneghar
Notario Público, hoy día 15 de Julio de 2004



SELLO

Victoria Eugenia Cruz de Ramirez
TRADUCTOR E INTERPRETE OFICIAL
RES. 201 - 22/04/02
MINISTERIO DE JUSTICIA

7

86
88

Firmado el día 17 de Agosto de 2004, en dos originales, por

(Firma)

(Cedente) Wen-Pin Scott Hsu

Firmada y Juramentada ante mí, Laurie Heneghar

Notario Público, hoy día 17 de Agosto de 2004

Laurie

Heneghar

Notario Público

SELLO

Firmado el día 23 de Agosto de 2004, en dos originales, por

MICROSOFT CORPORATION

Representada por (Firma)

(Cesionario)

David Bartley Eppenauer

Vicesecretario Corporativo

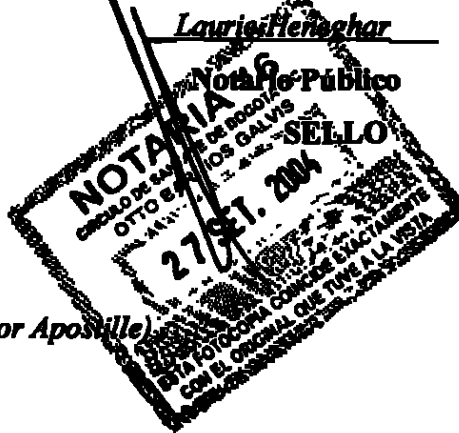
Firmada y Juramentada ante mí, Laurie Heneghar

Notario Público, hoy día 23 de Agosto de 2004

Laurie Heneghar

Notario Público

SELLO



(Este documento debe ser debidamente legalizado por Apostille)

Victoria Eugenia Cruz de Aguirre
TRADUCTOR E INTERPRETE OFICIAL
RES. 381 - 22/04/02
MINISTERIO DE JUSTICIA

20

89

CERTIFICACIÓN

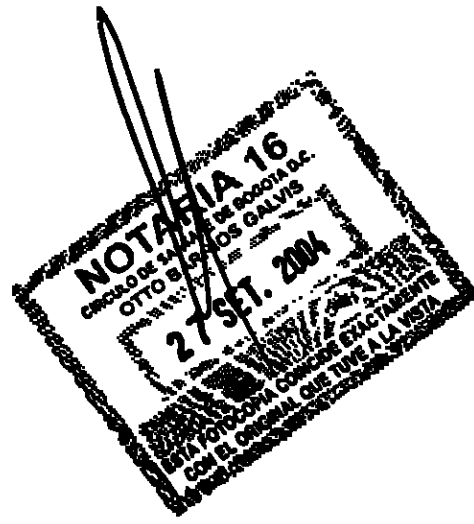
Yo, VICTORIA EUGENIA CRUZ DE RAMÍREZ, declaro que soy Traductora e Intérprete Oficial los idiomas Inglés y Español, con sede en la Calle 148 # 14-09 (202) en Bogotá DC, Colombia.

Declaro que la traducción adjunta es una versión fiel y completa al español del documento CESIÓN, contrato en Idioma Inglés, distinguido con el No.305442.23 y autenticado ante LAURIE HENEGHAR, Notario Público del Estado de Washington, Estados Unidos de América.

Victoria Eugenia Cruz de Ramírez
TRADUCTOR E INTERPRETE OFICIAL
RES. 381 - 22/04/02
MINJUSTICIA

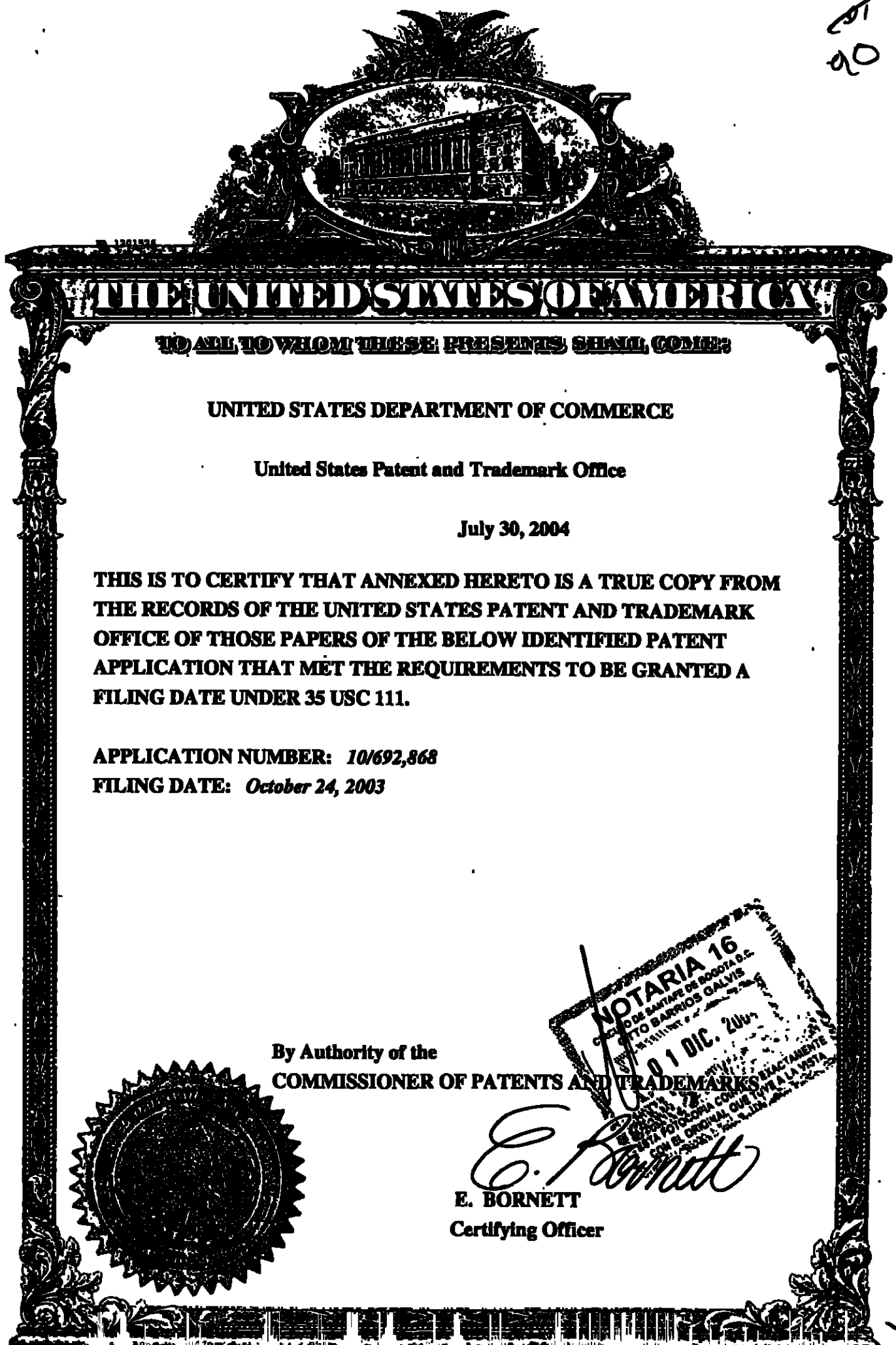


VICTORIA EUGENIA CRUZ DE RAMÍREZ
Traductor e Intérprete Oficial
Resol. 0391-22/04/02 de MinJusticia



27

20



THE UNITED STATES OF AMERICA

TO ALL TO WHOM THESE PRESENTS SHALL COME:

UNITED STATES DEPARTMENT OF COMMERCE

United States Patent and Trademark Office

July 30, 2004

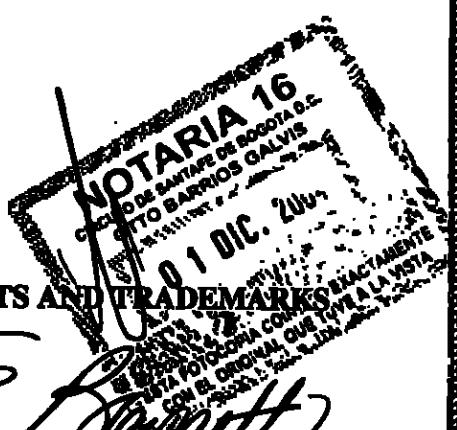
THIS IS TO CERTIFY THAT ANNEXED HERETO IS A TRUE COPY FROM THE RECORDS OF THE UNITED STATES PATENT AND TRADEMARK OFFICE OF THOSE PAPERS OF THE BELOW IDENTIFIED PATENT APPLICATION THAT MET THE REQUIREMENTS TO BE GRANTED A FILING DATE UNDER 35 USC 111.

APPLICATION NUMBER: 10/692,868
FILING DATE: October 24, 2003

**By Authority of the
COMMISSIONER OF PATENTS AND TRADEMARKS**



E. Bornett
E. BORNETT
Certifying Officer



7

DOCKET NO.: MSFT-2822/305442.1

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In Re Application of:

Caglar Gunyakti, Wen-Pin Scott Han,

Ning Zhang

For: PROGRAMMING INTERFACE FOR LICENSING

EXPRESS MAIL LABEL NO: EL 970382875 US
DATE OF DEPOSIT: October 24, 2003

EL 970382875US

☒ Mail Stop Patent Application ☐ Mail Stop Design ☐ Mail Stop Sequence
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

PATENT APPLICATION TRANSMITTAL LETTER

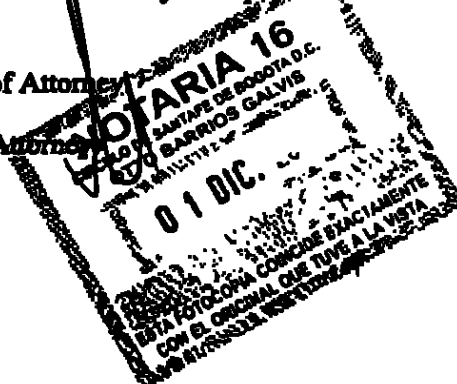
Transmitted herewith for filing, please find:

☒ A Utility Application under 37 CFR § 1.53(b).

The above-identified application includes the following:

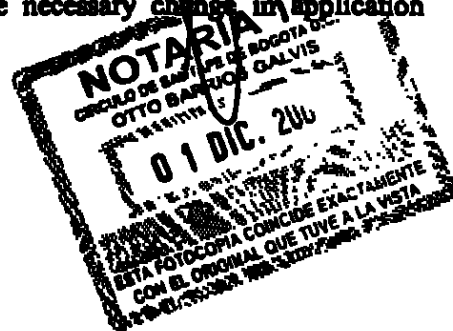
- ☐ A Request for Nonpublication
- ☒ A Specification including claims and abstract on pages 1 to 35.
- ☐ A copy of the specification including claims and abstract (pages -) and drawings and appendixes (if any) of earlier Application No. filed to which no new matter has been added. Such earlier application is hereby incorporated into the present application by reference.
- ☐ A New Unexecuted Declaration or Oath and Power of Attorney
- ☒ A New Executed Declaration or Oath and Power of Attorney

Page 1 of 5



22389 U.S. PTO
10/692868
102403

- ☐ A copy of the executed oath or declaration filed in the earlier application identified above is attached.
- ☐ An Associate Power of Attorney executed by _____ who is listed in the copy of the executed Declaration and Power of Attorney filed herewith.
- ☐ A Copy of the Power of Attorney with Revocation filed in prior Application No. _____
- ☐ Signed Statement deleting inventor(s) named in the prior application.
- ☐ This application claims foreign priority under 35 U.S.C. § 119 of Application No. _____ filed in _____ (Country).
- ☐ A Certified Copy of each of the above applications for which priority is claimed:
- ☐ is enclosed.
- ☐ has been filed in prior Application No. _____ filed _____
- ☒ Five (5) Sheets of Formal Drawings and/or Photographs.
- ☒ Figure 3 should be published.
- ☐ Petition to Accept Color Photographs is enclosed.
- ☐ This continuation/divisional application is assigned of record to _____ as evidenced by the assignment recorded on _____ at Reel No. _____ and Frame No. _____
- ☒ A newly Executed Assignment transferring rights to Microsoft Corporation.
- ☒ A Recordation Form Cover Sheet.
- ☒ Recordation Fee - \$40.00.
- ☐ A Preliminary Amendment is enclosed.
- ☐ A Sequence Listing consisting of pages 1- _____
- ☐ Diskette containing Sequence Listing is enclosed.
- ☐ The computer readable form in the above-identified application is identical with that filed in Application Number _____, filed _____. In accordance with 37 CFR § 1.821(e), please use the sequence listing filed on _____ and received by the PTO on _____ as the computer readable form filed in this application as the computer readable form for the instant application. It is understood that the Patent and Trademark Office will make the necessary changes in Application _____



number and filing date for the computer readable form that will be used for the instant application.

- ☐ A paper copy of the Sequence Listing:
- ☐ was filed in the above-identified parent application on _____ and received by the PTO on _____
- ☐ is included herewith in a separately filed preliminary amendment for incorporation into the specification.
- ☐ A Statement to Support Submission of Sequence Information is enclosed.
- ☐ Information Disclosure Statement including:
- ☐ Form 1449.
- ☐ Copies of References _____ listed on the attached Form PTO-1449.
- ☐ A copy of Petition for Extension of Time as filed in the prior case for copendency.
- ☐ Appended Material as follows:
- ☒ Return Receipt Postcard (should be specifically itemized).



DOCKET NO.: MSFT-2822/305442.1

PATENT

FEE CALCULATION:

- ☐ Applicant(s) by its/their undersigned attorney, claims small entity status under 37 CFR § 1.27 as (CHOOSE ONE: an Independent Inventor, a Small Business Concern, or a Nonprofit Organization).

			SMALL ENTITY		NOT SMALL ENTITY	
			RATE	FEE	RATE	FEE
DESIGN APPLICATION BASE FEE			\$170.00		\$340.00	
UTILITY APPLICATION BASE FEE			\$325.00		\$770.00	\$770.00
UTILITY APPLICATION; ALL CLAIMS CALCULATED AFTER ENTRY OF ALL AMENDMENTS						
	No. Filed	No. Extra				
TOTAL NO. OF CLAIMS	25-20 =	5	\$9 each		\$18 each	\$ 90.00
NO. OF INDEPENDENT CLAIMS	3-3 =	0	\$43 each		\$86 each	
MULTIPLE DEPENDENT CLAIM FEE			\$145		\$290	
ADDITIONAL FILING FEE						
TOTAL FILING FEE DUE						\$860.00

- ☒ Two checks are enclosed: (1) in the amount of \$860.00 (Application filing fee); and (2) in the amount of \$40.00 (Assignment filing fee). Please charge any deficiency or credit any overpayment to Deposit Account No. 23-3050.
- ☐ Please charge Deposit Account No. 23-3050 in the amount of \$ _____.00. This sheet is attached in duplicate.
- ☒ The Commissioner is hereby requested to grant an extension of time for the appropriate length of time, should one be necessary, in connection with this filing or any future filing submitted to the U.S. Patent and Trademark Office in the above-identified application during the pendency of this application. The Commissioner is further authorized to charge any fees related to any such extension of time to Deposit Account No. 23-3050. This sheet is provided in duplicate.
- ☒ Address all correspondence to Customer No. 23377.



23377

PATENT TRADEMARK OFFICE

Page 4 of 5



28

DOCKET NO.: MSFT-2822/305442.1

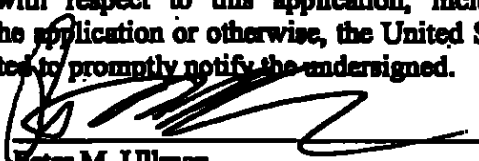
PATENT

☒ Direct all telephone calls to:

Peter M. Ullman
WOODCOCK WASHBURN LLP
One Liberty Place - 46th Floor
Philadelphia PA 19103
Telephone No.: (215) 568-3100
Facsimile No.: (215) 568-3439

SHOULD ANY DEFICIENCIES APPEAR with respect to this application, including deficiencies in payment of fees, missing parts of the application or otherwise, the United States Patent and Trademark Office is respectfully requested to promptly notify the undersigned.

Date: October 24, 2003


Peter M. Ullman
Registration No. 43,963

Page 5 of 5



96

PROGRAMMING INTERFACE FOR LICENSING

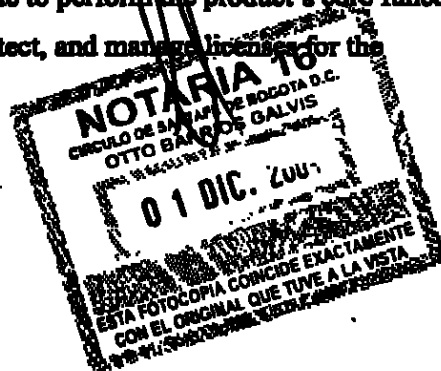
FIELD OF THE INVENTION

[0001] The present invention relates generally to the field of computer software, and, more particularly, to a programming interface that supports the enforcement of electronic licenses.

BACKGROUND OF THE INVENTION

[0002] Commercially-produced software has traditionally been made available under a license that defines the permissible terms of the software's use. When the practice of software licensing first began, the license generally took the form of a legal document that defined the user's rights with respect to the software. Such a document relied upon the legal system for enforcement. It has since become desirable for licenses to be enforced electronically - i.e., it is desirable that a computer program contain code that actively discourages or prevents use of the software in a manner that is contrary to the license.

[0003] Most software that provides for electronic license enforcement provides its own infrastructure to manage the licensing of the software and the use of the licenses. Thus, a typical commercial software product may include not only the code to perform the product's core function, but may also carry with it the code to obtain, evaluate, protect, and manage licenses for the



28

software. For each software vendor to develop and incorporate this infrastructure into its software is often a wasteful duplication of effort. It is therefore desirable to provide a system that performs the basic functions related to software licensing, where the system can be used by a broad variety of software applications in a uniform and defined way.

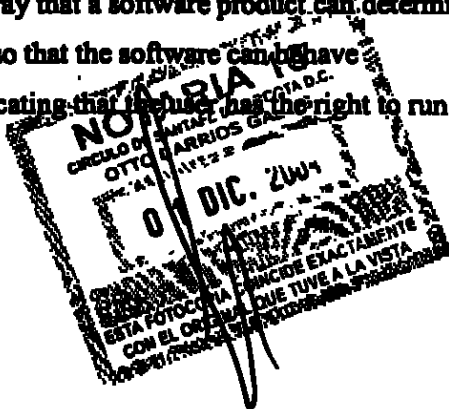
[0004] In view of the foregoing, there is a need for a mechanism that overcomes the drawbacks of the prior art.

SUMMARY OF THE INVENTION

[0005] The present invention provides a software licensing Application Programming Interface (API) that provides certain licensing functions for use by software products. A license service performs functions relating to the use of licenses, and exposes these functions to software products through the API. The service performs functions such as obtaining licenses, storing and managing licenses, protecting licenses from tampering, evaluating a license's validity, and evaluating whether a license is correctly bound to the machine and/or software product on which it is used. The software is able to make use of this functionality by calling the methods of the API.

[0006] In a typical use of the API, a software product calls an "open" API method in order to obtain a unique handle that is used by the license service to identify the application. The software product then calls a "consume right" API method. "Consume," in this context, means the exercise of a specified right. The call to the "consume right" method is parameterized by the software product's handle, and by the name of the right to be consumed. The license service then attempts to locate one or more valid, correctly bound licenses that contains the named right. If no such license exists, then the software product is notified of the failure. If such licenses exist, then the right is bound to one of the licenses, and the calling software product is notified of the binding. In such a case, the software product knows that the right exists, and can perform whatever functions are associated with this right.

[0007] In a preferred embodiment, the license service does not define what the software can or cannot do under the right, or enforce substantive constraints on the use of the software. Rather, the license service manages the licenses in such a way that a software product can determine by calling the API whether a right does, or does not, exist, so that the software can behave accordingly. For example, a right may be called "run," indicating that the software has the right to run



19

the software product. The software product can use the API to determine whether there is a valid (and correctly bound, and non-expired) right to run the software. However, if the API call returns with an indication that there is no right to run the software, it is up to the software to cease operation or take some other action based on the non-existence of this right.

[0008] A right may be associated with information, which becomes available after a successful call to the "consume right" method. For example, a given software product may have individual rules about when it is permissible to edit, print, save, etc., and these rules can be stored in the license that contains the right. The API provides a "get information" method that allows this information to be retrieved from the license.

[0009] . Other features of the invention are described below.

BRIEF DESCRIPTION OF THE DRAWINGS

[0010] The foregoing summary, as well as the following detailed description of preferred embodiments, is better understood when read in conjunction with the appended drawings. For the purpose of illustrating the invention, there is shown in the drawings exemplary constructions of the invention; however, the invention is not limited to the specific methods and instrumentalities disclosed. In the drawings:

[0011] FIG. 1 is a block diagram of an example computing environment in which aspects of the invention may be implemented;

[0012] FIG. 2 is a block diagram of an architecture in which a system performs licensing functions and exposes an API for use by software products;

[0013] FIG. 3 is a flow diagram of a method through which a software product uses a licensing API;

[0014] FIG. 4 is a flow diagram of a method by which a software product consumes a right; and

[0015] FIG. 5 is a flow diagram of a method by which a software product retrieves information relating to a consumed right.

DETAILED DESCRIPTION OF THE INVENTION



20

Overview

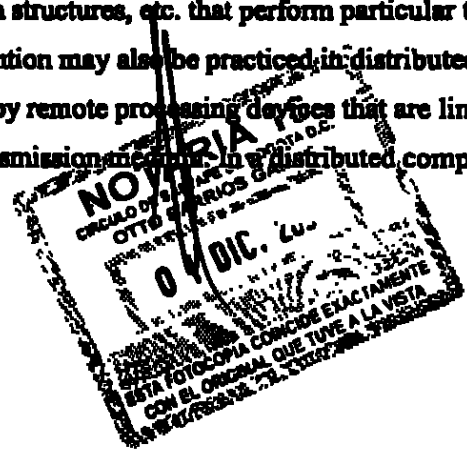
[0016] The use of commercial software is typically governed by a license, and it has become increasingly common for this license to be embodied in an electronic form that can be enforced by the software itself. One challenge in creating an electronic licensing system is that an infrastructure is needed to manage the use of the licenses. Replicating this infrastructure for every software product is cumbersome and wasteful. The present invention provides an API that allows different software products to use a common infrastructure that performs various licensing functions.

Exemplary Computing Arrangement

[0017] FIG. 1 shows an exemplary computing environment in which aspects of the invention may be implemented. The computing system environment 100 is only one example of a suitable computing environment and is not intended to suggest any limitation as to the scope of use or functionality of the invention. Neither should the computing environment 100 be interpreted as having any dependency or requirement relating to any one or combination of components illustrated in the exemplary operating environment 100.

[0018] The invention is operational with numerous other general purpose or special purpose computing system environments or configurations. Examples of well known computing systems, environments, and/or configurations that may be suitable for use with the invention include, but are not limited to, personal computers, server computers, hand-held or laptop devices, multiprocessor systems, microprocessor-based systems, set top boxes, programmable consumer electronics, network PCs, minicomputers, mainframe computers, embedded systems, distributed computing environments that include any of the above systems or devices, and the like.

[0019] The invention may be described in the general context of computer-executable instructions, such as program modules, being executed by a computer. Generally, program modules include routines, programs, objects, components, data structures, etc. that perform particular tasks or implement particular abstract data types. The invention may also be practiced in distributed computing environments where tasks are performed by remote processing devices that are linked through a communications network or other data transmission medium. In a distributed computing

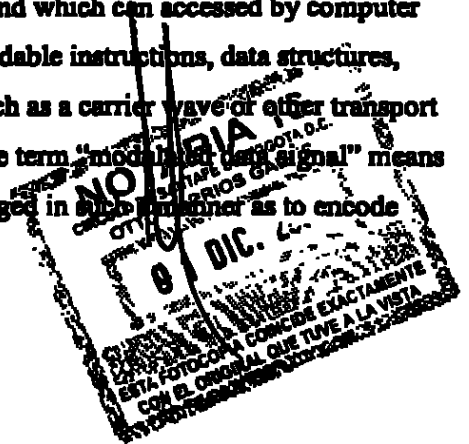


21

environment, program modules and other data may be located in both local and remote computer storage media including memory storage devices.

[0020] With reference to FIG. 1, an exemplary system for implementing the invention includes a general purpose computing device in the form of a computer 110. Components of computer 110 may include, but are not limited to, a processing unit 120, a system memory 130, and a system bus 121 that couples various system components including the system memory to the processing unit 120. The processing unit 120 may represent multiple logical processing units such as those supported on a multi-threaded processor. The system bus 121 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. By way of example, and not limitation, such architectures include Industry Standard Architecture (ISA) bus, Micro Channel Architecture (MCA) bus, Enhanced ISA (EISA) bus, Video Electronics Standards Association (VESA) local bus, and Peripheral Component Interconnect (PCI) bus (also known as Mezzanine bus). The system bus 121 may also be implemented as a point-to-point connection, switching fabric, or the like, among the communicating devices.

[0021] Computer 110 typically includes a variety of computer readable media. Computer readable media can be any available media that can be accessed by computer 110 and includes both volatile and nonvolatile media, removable and non-removable media. By way of example, and not limitation, computer readable media may comprise computer storage media and communication media. Computer storage media includes both volatile and nonvolatile, removable and non-removable media implemented in any method or technology for storage of information such as computer readable instructions, data structures, program modules or other data. Computer storage media includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CDROM, digital versatile disks (DVD) or other optical disk storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store the desired information and which can be accessed by computer 110. Communication media typically embodies computer readable instructions, data structures, program modules or other data in a modulated data signal such as a carrier wave or other transport mechanism and includes any information delivery media. The term "modulated data signal" means a signal that has one or more of its characteristics set or changed in such a manner as to encode

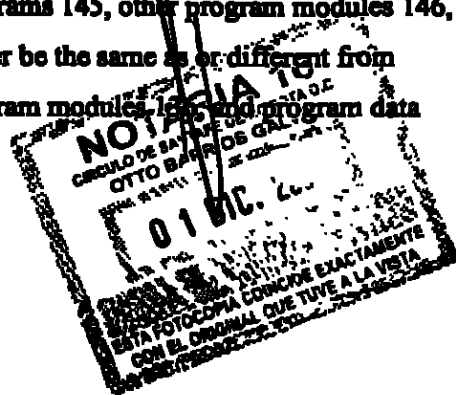


information in the signal. By way of example, and not limitation, communication media includes wired media such as a wired network or direct-wired connection, and wireless media such as acoustic, RF, infrared and other wireless media. Combinations of any of the above should also be included within the scope of computer readable media.

[0022] The system memory 130 includes computer storage media in the form of volatile and/or nonvolatile memory such as read only memory (ROM) 131 and random access memory (RAM) 132. A basic input/output system 133 (BIOS), containing the basic routines that help to transfer information between elements within computer 110, such as during start-up, is typically stored in ROM 131. RAM 132 typically contains data and/or program modules that are immediately accessible to and/or presently being operated on by processing unit 120. By way of example, and not limitation, FIG. 1 illustrates operating system 134, application programs 135, other program modules 136, and program data 137.

[0023] The computer 110 may also include other removable/non-removable, volatile/nonvolatile computer storage media. By way of example only, FIG. 1 illustrates a hard disk drive 140 that reads from or writes to non-removable, nonvolatile magnetic media, a magnetic disk drive 151 that reads from or writes to a removable, nonvolatile magnetic disk 152, and an optical disk drive 155 that reads from or writes to a removable, nonvolatile optical disk 156, such as a CD ROM or other optical media. Other removable/non-removable, volatile/nonvolatile computer storage media that can be used in the exemplary operating environment include, but are not limited to, magnetic tape cassettes, flash memory cards, digital versatile disks, digital video tape, solid state RAM, solid state ROM, and the like. The hard disk drive 141 is typically connected to the system bus 121 through a non-removable memory interface such as interface 140, and magnetic disk drive 151 and optical disk drive 155 are typically connected to the system bus 121 by a removable memory interface, such as interface 150.

[0024] The drives and their associated computer storage media discussed above and illustrated in FIG. 1, provide storage of computer readable instructions, data structures, program modules and other data for the computer 110. In FIG. 1, for example, hard disk drive 141 is illustrated as storing operating system 144, application programs 145, other program modules 146, and program data 147. Note that these components can either be the same as or different from operating system 134, application programs 135, other program modules 136, and program data 137.

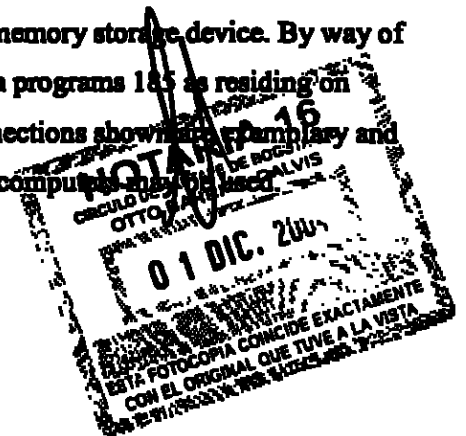


73

137. Operating system 144, application programs 145, other program modules 146, and program data 147 are given different numbers here to illustrate that, at a minimum, they are different copies. A user may enter commands and information into the computer 20 through input devices such as a keyboard 162 and pointing device 161, commonly referred to as a mouse, trackball or touch pad. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 120 through a user input interface 160 that is coupled to the system bus, but may be connected by other interface and bus structures, such as a parallel port, game port or a universal serial bus (USB). A monitor 191 or other type of display device is also connected to the system bus 121 via an interface, such as a video interface 190. In addition to the monitor, computers may also include other peripheral output devices such as speakers 197 and printer 196, which may be connected through an output peripheral interface 195

[0025] The computer 110 may operate in a networked environment using logical connections to one or more remote computers, such as a remote computer 180. The remote computer 180 may be a personal computer, a server, a router, a network PC, a peer device or other common network node, and typically includes many or all of the elements described above relative to the computer 110, although only a memory storage device 181 has been illustrated in FIG. 1. The logical connections depicted in FIG. 1 include a local area network (LAN) 171 and a wide area network (WAN) 173, but may also include other networks. Such networking environments are commonplace in offices, enterprise-wide computer networks, intranets and the Internet.

[0026] When used in a LAN networking environment, the computer 110 is connected to the LAN 171 through a network interface or adapter 170. When used in a WAN networking environment, the computer 110 typically includes a modem 172 or other means for establishing communications over the WAN 173, such as the Internet. The modem 172, which may be internal or external, may be connected to the system bus 121 via the user input interface 160, or other appropriate mechanism. In a networked environment, program modules depicted relative to the computer 110, or portions thereof, may be stored in the remote memory storage device. By way of example, and not limitation, FIG. 1 illustrates remote application programs 185 as residing on memory device 181. It will be appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computers may be used.



103

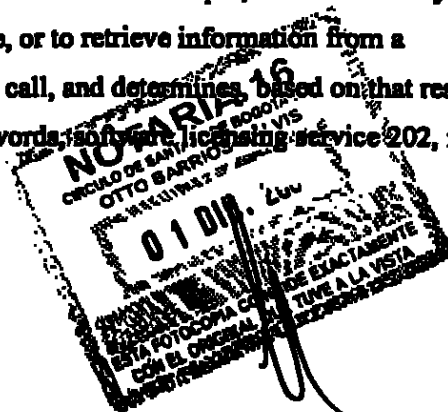
Software Licensing Service

[0027] FIG. 2 shows an example system that provides a software licensing service 202. Software licensing service 202 operates inside of computer 110 (shown in FIG. 1). In one example, software licensing service 202 is part of an operating system that executes on computer 110. Software licensing service maintains a license store 204 in which license files for software are stored. License files may, for example, be eXtensible Rights Markup Language (XrML) files that specify rights to software, and that also may specify various types of conditions on the exercise of those rights. Software 204 also maintains a trust store 206. Trust store 206 stores un-authenticatable, dynamic data in a tamper-resistant manner; trust store 206 stores data that is used in the license validation process. For example, certain licenses may have expiration dates, and, in order to prevent the expiration data from being circumvented through clock rollback, the current time (and elapsed time) may be periodically stored in trust store 206 to ensure that the clock is always moving forward.

[0028] Software licensing service 202 manages license store 204 and trust store 206, and also performs various functions relating to the licensing of software. For example, software licensing service 202 may contain modules that parse license files, modules that enforce the binding of a license to a particular machine and/or to a particular instance of a software product, and a secure timer/counter (that uses trust store 206 in the manner described above).

[0029] Software licensing service 202 exposes an application programming interface (API) 208 that allows application software (such as application 135) to use software licensing service 202. Application 135 may invoke the features of software licensing service 208 by making local procedure call (LPC) to the methods of API 208. An example set of API methods that may be exposed by software licensing service 202 is described below.

[0030] The manner in which API 208 is used by an application is described with reference to FIG. 3. Initially, the application makes an API call (302). Service 202 then processes the API call (304), and returns the result of the API call to the application 306. For example, an API call may request to exercise ("consume") a right granted in a license, or to retrieve information from a license. The application then receives the result of the API call, and determines, based on that result, what the application's behavior should be (308). In other words, software licensing service 202, in a



23

preferred embodiment, does not enforce a license directly, but rather provides the infrastructure through which licenses can be managed and used. For example, if an application makes an API call to consume a right and service 202 determines that there is no valid license granting this right, service 202, in a preferred embodiment, does not prevent the application from running, but instead informs the application that the right is not available. Thus, the application can use its own mechanisms to determine what to do in response to the unavailability of the right. This facet of the API provides software vendors the flexibility to decide how the licensing infrastructure provided by service 202 should be used. In another implementation, application can be bound to licensing service functionality.

Example Software Licensing API

[0031] The following is an example set of API methods that may be exposed by a software licensing service:

SLOpen

[0032] The SLOpen function opens a SL client context handle that must be used for all subsequent SL API calls. (Throughout the example API descriptions, "SL" shall refer to the software licensing service. Thus, the SL client context handle is the handle used by a client in communicating with the software licensing service.)

RESULT

SLOpen(

CONST GUID* *pguidApp*,

HSLC* *phSLC*

);

Parameters

pguidApp

[0033] [in] Pointer to application GUID that uniquely identifies an application. If this argument is NULL, an E_INVALIDARG error is returned.

phSLC

[0034] [out] SL client context Handle or INVALID_HANDLE_VALUE.



Remarks

[0035] Use the SLClose function to close an context handle returned by SLOpen.

[0036] Application GUID: Unique ID of application. For the WINDOWS version of the MICROSOFT OFFICE application suite, WinWord has an Application GUID which is different from Excel's Application GUID. For Windows, Windows itself is an application, although it is a composite of many programs.

[0037] In Office case, A user can install both Office Suite and WinWord standalone products on the machine. From SL point of view, the WinWord in both products has same Application GUID. WinWord's Application GUID associated to two Product GUID. In other words, WinWord can use either Office Suite's product license or WinWord's product license.

[0038] When SLOpen succeeds:

[0039] A RPC (remote procedure call) binding has been established.

[0040] A context memory is created on SL service. The context is used to keep the status information for the caller of the client.

[0041] The SLC handle is like a file handle. A process can open multiple SL context handles but handles are valid within the caller process.

Returns

[0042] Success or failure

SLClose

[0043] The SLClose function closes an opened SL client context handle. Any information in the context is released automatically.

HRESULT

SLClose(

HSLC hSLC

);

Parameters



MSFT-2822/305442.01

hSLC

[0044] [in] Handle to current SL client context handle

Remarks

[0045] When SLClose is done, the RPC binding is released, and the context is destroyed.

Returns

[0046] Success or failure.

SLInstall

[0047] The SLInstall function installs applications' licenses and registers applications' information.

```
HRESULT SLInstall(  
    HSLC          hSLC,  
    CONST SL_PRODKEY* pAppPrdKey,  
    DWORD          dwNumOfApps,  
    CONST GUID*     pguidApps,  
    DWORD          dwNumOfLicFiles,  
    PCWSTR          ppszLicFiles[],  
    BOOL            bVerify  
);
```

Parameters

hSLC

[0048] [in] Handle to current SL client context handle

pAppPrdKey

[0049] [in] Application product key structure. The product key can be Microsoft product key format or Application's product key format.

```
typedef struct _tagSL_PRODKEY  
{
```



MSFT-2822/305442.01

108
107

```
        DWORD cbSize;                                // Size of SL_PRODKEY
structure
        DWORD dwVersion;                            // Version of SL_PRODKEY structure
        WCHAR szProdKey[MAX_PRODKEYSTR_SIZE+1];
        SL_PRODKEY_TYPE eProdKeyType;                // Type of Product key
        SL_CUSTOM_PRODKEY_INFO CustomPrdKeyInfo;      // Customer product key info
    } SL_PRODKEY;
```

The eProdKeyType can be one of following values:

SL_PRODKEY_CUSTOM

SL_PRODKEY_MS2002

SL_PRODKEY_MS2003

[0050] If the Product Key type is non-MS Product Key (i.e. eProdKeyType == SL_PRODKEY_CUSTOM), the caller has to fill in its custom product key information. If the product uses MS Product Key, then CustomPrdKeyInfo can be ignored.

```
typedef struct _tagSL_CUSTOM_PRODKEY_INFO
        DWORD dwSKUID;                                // Unique ID for specific SKU, for example
Group ID in MS PID.
        DWORD dwSerialNumber;                        // unique serial number, e.g. channel + sequence
number in MS PID.
    } SL_CUSTOM_PRODKEY_INFO;
```

[0051] Current version number of SL_PRODKEY is 1. The caller can use SL_CURRENT_PRODKEY_VERSION in dwVersion field.

dwNumOfApps

[0052] [in] The number of application GUID in pguidApps



27

pguidApps

[0053] [in] A list of application of GUID. The application GUID represents the application that the license is being installed for. For example, Office Setup program can call this function to install the license(s) for Word, Excel by specifying each application GUID in *pguidApps*. *pguidApp* cannot be NULL here.

dwNumOfApps

[0054] [in] The number of license files.

ppszLicFile

[0055] [in] File names in array of strings .

Returns

[0056] Success or failure

SLUninstall

[0057] The SLUninstall function uninstalls a product's license from an application.

HRESULT SLUninstall (

HSLC hSLC,

CONST SL_PRODKEY* pAppPrdKey

);

Parameters

hSLC

[0058] [in] Handle to current SL client context handle

pAppPrdKey

[0059] [in] See above definition in connection with SLInstall

Remarks



[0060] An application could have more than one product licenses. For example, when the user uninstalls Office suite, the association between Office Suite license and WinWord should be removed, but the license from WinWord Standalone product should not be removed.

[0061] When SLUninstall succeeds:

[0062] The information associated with this Product Key is removed. (see SLInstall, above, for the associated information)

[0063] The product keys associated with the Application GUID is removed.

[0064] The license files associated with the product GUID are preferably still kept.

Returns

[0065] Success or failure

SLConsumeRight

[0066] The SLConsumeRight function lets an application to examine or exercise the rights on a locally-stored license. Calling this function binds a license to the right mentioned in *pszRightName*. If this right cannot be exercised by the current caller, then the application fails. If the function succeeds, the action associated with the right can be executed (like decreasing usage count, decreasing time quota, or nothing)

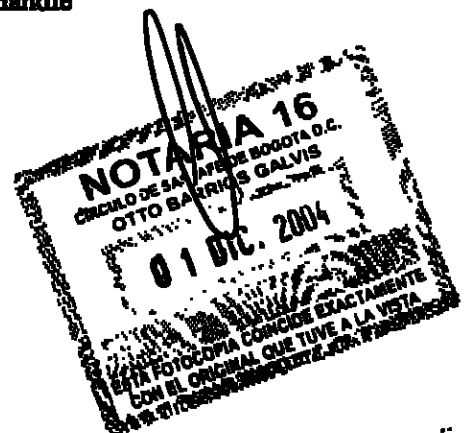
```
HRESULT SLConsumeRight(  
    HSLC          hSLC,  
    PCWSTR        pszRightName,  
    SL_ASYNC_CONTEXT* pAsyncContext  
);
```

Parameters

hSLC

[0067] [in] Handle to current SL client context handle

pszRightName



[0068] [in] The name of right needs to be evaluated. In current design, the right name is defined by applications. SL opens the license and evaluates the condition based on the right name.

pAsyncContext

[0069] [in/out] If *pAsyncContext* is NULL, this function works in synchronous mode, otherwise, the function works in asynchronous mode. SL_ASYNC_CONTEXT is opaque to caller and managed by SLC.

Remarks

[0070] All licenses associated with the application GUID (specified in SLOpen) will be conceptually combined in one logic license.

[0071] If there are multiple consumable grants of the right, then the license with higher priority will be consumed first.

Returns

[0072] Success or failure

SLInitializeAsyncContext

[0073] The SLInitializeAsyncContext function initializes the asynchronous context for SLC functions to make asynchronous call.

HRESULT SLInitializeAsyncContext(

SL_ASYNC_CONTEXT* *pAsyncContext*, // asynchronous context

HANDLE *hEvent*, // event handle

PVOID *pvReserved* // reserved, NULL

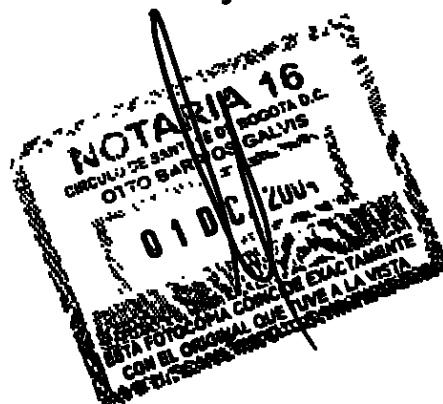
);

Parameters

pAsyncContext

[0074] [in/out] pointer to asynchronous context that contains asynchronous call information..

hEvent



32

[0075] [in] The event object used for synchronization.

pvReserved

[0076] [in] reserved for extension.

Returns

[0077] Success or failure.

SLCancelAsyncCall

[0078] The SLCancelAsyncCall function is used to cancel an asynchronous call.

HRESULT SLCancelAsyncCall(

SL_ASYNC_CONTEXT* *pAsyncContext*, // asynchronous context

BOOL *fAbortCall* // cancel immediately

);

Parameters

pAsyncContext

[0079] [in] asynchronous context for SL asynchronous call.

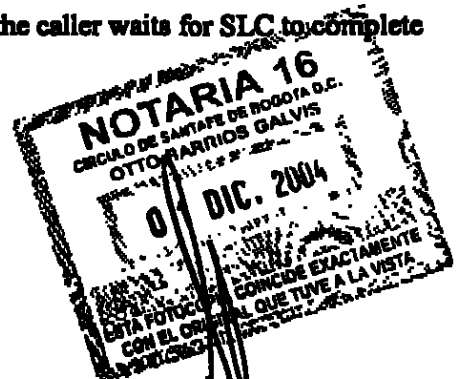
fAbortCall

[0080] [in] If TRUE, the call is cancelled immediately. If FALSE, wait for the SL to complete the call.

Remarks

[0081] There are two ways for the caller to request cancellation of an asynchronous call—abortive and nonabortive. In an abortive cancel (*fAbortCall* is TRUE), the SLCancelAsyncCall function sends a cancel notification to the SLC and the asynchronous call is canceled immediately, without waiting for a response from the SLC. In a nonabortive cancel (*fAbortCall* is FALSE) the SLCancelAsyncCall function notifies SLC of the cancel and the caller waits for SLC to complete the call.

Returns



[0082] Success or failure.

SLCompleteAsyncCall

[0083] The SLCompleteAsyncCall function is used to complete an SLC asynchronous call.

```
HRESULT SLCompleteAsyncCall(  
    SL_ASYNC_CONTEXT*    pAsyncContext,    // asynchronous context  
    HRESULT*              phrAsyncCall      // error code of the submitted  
asynchronous call  
);
```

Parameters

pAsyncContext

[0084] [in] asynchronous context for SL asynchronous call.

phrAsyncCall

[0085] [out] the error code of the submitted asynchronous call.

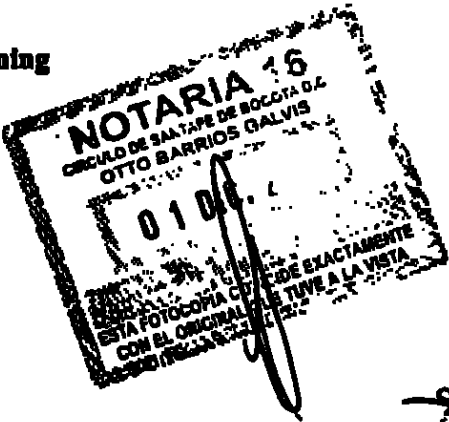
Remarks

[0086] If the caller calls this function before the reply has arrived, the call returns E_SLC_ASYNC_CALL_PENDING. The buffer must be valid and it must be big enough to receive the return value. If the call does not return E_SLC_ASYNC_CALL_PENDING, this SLCompleteAsyncCall invocation is final for the asynchronous call. After this function call, regardless of success or failure, all resources allocated for this asynchronous call are freed. (Subsequent calls to the SLCompleteAsyncCall or SLCancelAsyncCall functions have undefined results until a new call on the SL_ASYNC_CONTEXT structure is initiated).

Returns

Value

Meaning



MSFT-2822/305442.01

S_OK	The call was completed successfully.
E_SLC_INVALID_ASYNC_CONTEXT	The asynchronous call context is not valid.
E_SLC_ASYNC_CALL_PENDING	The call has not yet completed.
E_SLC_CALL_CANCELLED	The call was cancelled.

SLGetInformation

[0087] The SLGetLicenseInfo function is used to get a variety of information.

HRESULT SLGetInformation(

HSLC *hSLC*, // SL client context handle
DWORD *dwCategory*, // The category of information to retrieve
PCWSTR *pszKeyName*, // Name of the Key
DWORD* *pdwType*, // Type of value
SIZE_T* *pcbValue*, // Size of value
FBYTE* *ppbValue* // Pointer to buffer of value
);

Parameters

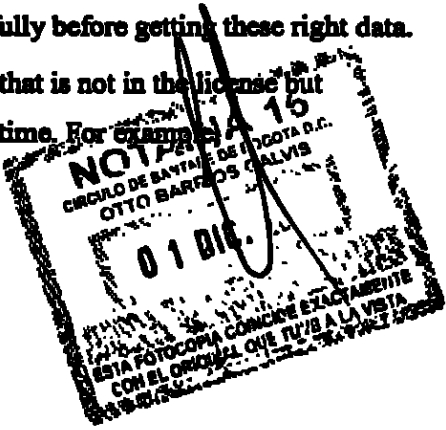
hSLC

[0088] [in] Handle to current SL client context handle

dwCategory

[0089] [in] The category of information.

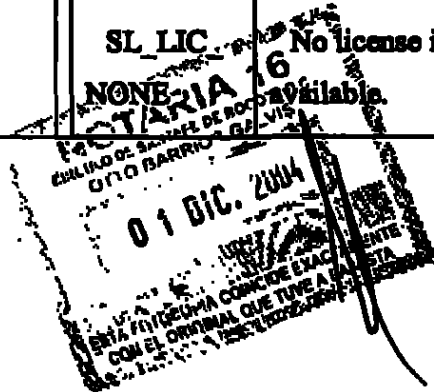
Category	Meaning
SL_CAT_RIGHTDATA	Get the information from bound right. The license has to be consumed successfully before getting these right data.
SL_CAT_DYNAMICPROPERTY	Get the information that is not in the license but calculating in the run-time. For example:



35

RemainingGracePeriodDays. The right has to be consumed before calling.

Name	Meaning						
RemainingGracePeriodDays : DWORD	The grace period is defined in out-of-box license. Once the application is installed, the time is counting down. Applications can check remaining grace period after they have consumed license.						
ActivationStatus : DWORD	After applications consumed license, it can get consumed license type. The return value could be: <table><tr><td>SL_LIC_OOB</td><td>The consumed license is out-of-box license.</td></tr><tr><td>SL_LIC_ACQUIRED</td><td>The consumed license is acquired license.</td></tr><tr><td>SL_LIC_NONE</td><td>No license is available.</td></tr></table>	SL_LIC_OOB	The consumed license is out-of-box license.	SL_LIC_ACQUIRED	The consumed license is acquired license.	SL_LIC_NONE	No license is available.
SL_LIC_OOB	The consumed license is out-of-box license.						
SL_LIC_ACQUIRED	The consumed license is acquired license.						
SL_LIC_NONE	No license is available.						



116
115

MSFT-2822/305442.01

SL_CAT_SERVICEINFO

Get the information that is not dependent on license. The caller can get this category of information without consuming license.

Name	Meaning
SLVersion: DWORD	The version of SL. 1.2.3.4 format.
HWID: BINARY	Current HWID

SL_CAT_WINDOWSINFO

Get information that is bound right property in Windows license. This is for Componentization. Windows License has been consumed by SL service and SL service keeps the bound right properties.

SL_CAT_ENUMLICINFO

When SLEnumLicense is called and is succeed, the caller can query the information of enumerated license by using this category.

pszKeyName

[0090] [in] the name of the key. E.g. BuildNumber

pdwType

[0091] [out] type of data

Value	Meaning
SL_DATATYPE_SZ	Unicode string
SL_DATATYPE_DWORD	DWORD
SL_DATATYPE_BINARY	Binary



pcbValue

[0092] [out] Size of the buffer allocated (in bytes).

ppbValue

[0093] [out] If successful, the data is returned in the buffer allocated by SLC. The caller has to call SLFreeMemory to free the memory.

Returns

[0094] Success or failure

SLAcquireLicense

[0095] The SLAcquireLicense function is used to acquire on-line license for the user. SLC enumerates the product keys associated with the Application and picks up the product key with highest product priority (see SLInstall, registration information). Then SL gets the clearing house URL from out-of-box license and connects to clearing house to get a license.

[0096] SLAcquireLicense could be a lengthy process. Applications can call this function in asynchronous mode by specifying *pAsyncContext* (NULL means synchronous mode).

HRESULT SLAcquireLicense{

HSLC *hSLC*, // SL client context handle

PCWSTR *pszProdKeyHash*, // hash of product key

PCWSTR *pszPublishLicense*, // string of publishing license.

SL_ASYNC_CONTEXT* *pAsyncContext*

};

Parameters***hSLC***

[0097] [in] Handle to current SL client context handle

pszProdKeyHash

MSFT-2822/305442.01

[0098] [in] string of product key hash. The product key hash is created when SLInstall is called and is maintained by the licensing service.

pszPublishingLicense

[0099] [in] string of publishing license.

pAsyncContext

[0100] [in] asynchronous context for SL asynchronous call.

Remarks

[0101] The acquired license will be stored in license store accordingly and the license information will be registered, too. (see SLInstall)

[0102] Applications might need to add more client information to license. The application can put the information to pbAppData in the call and this data will be sent to clearing house.

[0103] When this function succeeds:

[0104] It sent necessary binding information to the specified license server.

[0105] It receives the license from license server.

[0106] It stored the license in license store. See description of SLInstall, above, for how the license file is stored.

Returns

[0107] Success or failure.

SLGenerateTextChallenge

[0108] Generates and installation challenge text to be routed to a license issuer in an out of band fashion (telephone, email, file share, etc).

HRESULT SLGenerateTextChallenge(

HSLC *hSLC*, // SL client context handle

PCWSTR *pszProdKeyHash*, // string of product key hash

- 22 -



39

MSFT-2822/305442.01

118

```
    BOOL    fSingleSession, // Single session only?
    PWSTR    *ppszChallenge // Pointer to buffer to hold challenge text
};
```

Parameters

hSLC

[0109] [in] Handle to current SL client context handle

pszProdKeyHash

[0110] [in] String of product key hash. The product key hash is created when SLInstall is called and maintained by the licensing service.

bSingleSession

[0111] [in] Specifies whether or not the corresponding text response from the license issue will be valid only for the lifetime of this SLC session handle.

ppszChallenge

[0112] [out] If successful, the text challenge is returned in the buffer allocated by SLC. The caller needs to call SLFreeMemory to free the allocated memory.

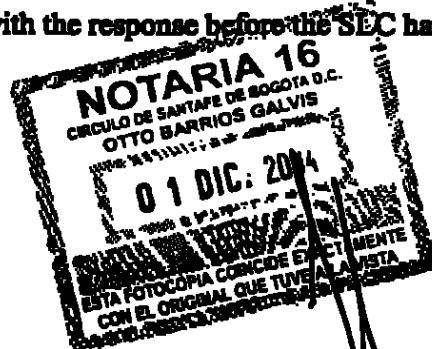
Returns

[0113] Success or failure.

SLDepositTextResponse

[0114] Deposits the response to an installation challenge text in the licensing system. Used to activate a license with a conditional access code. Only valid if there is an outstanding text challenge which has been issued for a license. If the original license specified that the challenge was only valid for a single session, this API must be called with the response before the SLC handle is closed or depositing the response will fail.

HRESULT SLDepositTextResponse(



MSFT-2822/305442.01

125
119

```
HSLC    hSLC,        // SL client context handle
PCWSTR  pszProdKeyHash,
PWSTR    pszResponse  // Buffer containing response text
};
```

Parameters

hSLC

[0115] [in] Handle to current SL client context handle

pszProdKeyHash

[0116] string of the product key hash

pszChallenge

[0117] [in] Response text

Returns

[0118] Success or failure.

SLEnumLicense

[0119] The SLEnumLicensefunction is used to enumerate installed licenses and get information from the license.

HRESULT SLEnumLicense(

```
HSLC    hSLC,        // SL client context handle
CONST GUID* pguidApp,  // application GUID
DWORD    dwIndex     // index number
);
```

Parameters

hSLC

[0120] [in] Handle to current SL client context handle



pguidApp

[0121] [in] See SLInstall. If *pguidApp* is not NULL, then licenses associated with this GUID are enumerated. If the GUID is NULL, then all licenses are enumerated.

dwIndex

[0122] [in] The index number of the license to be retrieved. This value should be zero for the first call to the SLEnumLicense function and then incremented for subsequent calls.

[0123] The function may return licenses in any order.

Returns

[0124] E_SL_NO_MORE_DATA - No license at the specified index.

Remarks

[0125] If SLEnumLicenses succeeded, the selected license information can be accessed by calling SLGetInformation and the category is SL_CAT_ENUMLICINFO

Sample:

DWORD i=0;

PBYTE pbProductPriority = NULL;

PBYTE pbRemainingGracePeriodDays = NULL;

for (i=0; ; i++)

{

EXIT_ON_ERROR(SLEnumLicense(hSLC, NULL, dwIndex));

if (E_SL_NO_MORE_DATA == SCODE(hr))

{

hr = S_OK;

break;



}

EXIT_ON_ERROR(SLGetInformation(hSLC, SL_CAT_ENUMLIC, "ProductPriority",
&dwType, &cbProductPriority, &pbProductPriority));

EXIT_ON_ERROR(SLGetInformation(hSLC, SL_CAT_ENUMLIC,
"RemainingGracePeriodDays", &dwType, &cbRemainingGracePeriodDays,
&pbRemainingGracePeriodDays));

Exit:

SLFreeMemory(pbProductPriority);
SLFreeMemory(pbRemainingGracePeriodDays);

}

SLFreeMemory

[0126] The SLFreeMemory function is used to free the memory allocated by SLC.

VOID SLFreeMemory(
 PVOID pvMemBlock, // pointer to memory
);

Parameters

pvMemBlock

[0127] [in] Previously allocated memory block to be freed

Returns

[0128] None

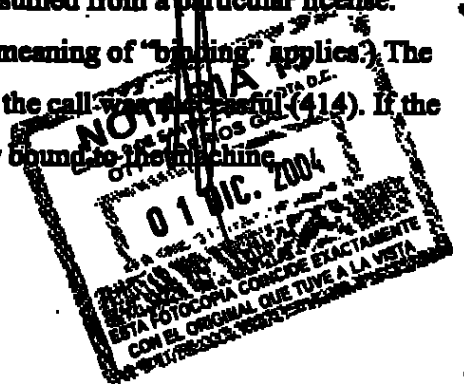


Use of Software Licensing API to Control Use of Software

[0129] A software product uses the API of the present invention for various purposes related to licensing, including the consumption of rights in a license, and the retrieval of data from the license. As noted above, the API allows the software to determine what rights are present in the license, but, preferably, it is up to the software to determine what to do with that information – e.g., grant or deny access to a feature, cease operation altogether, etc. The following description of FIGS. 4 and 5 show how the API of the present invention is used by a software product.

[0130] FIG. 4 shows an example process by which an application “consumes” a right. The application calls the SLConsumeRight method (402). As discussed above, the arguments to the SLConsumeRight function include the client handle assigned by the licensing service, and the name of the right (which is assigned by the vendor of the software to which the right pertains). The licensing service (service 202, shown in FIG. 2) receives the call (404). The service then locates licenses that contains the right, and checks the licenses bindings and validity. As noted above, the license is located in the license store; if there is more than one license that pertains to the application software to which the SLConsumeRight call pertains, then a priority rule may be used to select one of the applicable licenses. Checking the binding means determining that: (1) the license is bound to the product key of the application identified by the client handle; and (2) the license is bound to the machine on which the software is running (or to the group of machines of which the current machine is a member). Checking validity may include determining that the right has not expired (in the case of licenses that specify an expiration date), and that the maximum number of uses of the right is not exceeded (in the case where the license specifies a maximum number of times that the right may be used (i.e., “consumed”)).

[0131] If the license and/or right are found to be correctly bound and valid (408), then the license is bound to the right requested in the API call (412). (It should be noted that “binding” a license to a machine, environment, and a product key means that the license specifies which machine(s) and product key it can be used with; “binding” a license to a right means that the consume function has been successful, and the right is being consumed from a particular license. Throughout this description, it will be clear from context which meaning of “binding” applies.) The API call then returns to the calling application and indicates that the call was successful (414). If the license and/or right has been found to be invalid, or not correctly bound to the machine



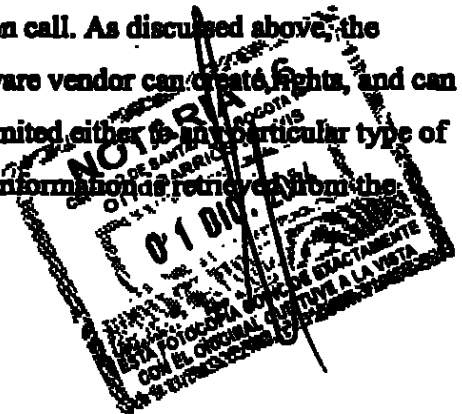
environment, or product ID, then the SLConsumeRight call returns to the calling application and indicates that the operation failed (410).

[0132] If the SLConsumeRight call returns with a failure, then the right specified in the call cannot be consumed from a license, and no information about that right will be available to the calling application. However, if the right is successfully consumed, then the application can use the binding of the right to the license to get information from the license about the right. For example, a license may contain a general right called "run," which indicates that the application may be run. However, for the "run" right, the license may contain more specific parameters about the usage of the application - e.g., the license may specify whether particular features of an application (e.g., print, edit, save, etc.) should be turned on or off, and may give specific parameters for the use of these features (e.g., the document can be saved only on machines that are running in a particular domain, or the print feature can only be used for thirty days, etc.). The SL API does not require any particular type of information to be associated with a right, but rather provides a mechanism whereby an application vendor can associate any type of information with a right, which can then be retrieved and interpreted by the application.

[0133] Assuming that a right has been successfully consumed as described in FIG. 4, the application may then retrieve the information associated with the right. The process of retrieving this information is described in FIG. 5.

[0134] First, the application calls the SLGetInformation method on the bound right (502). The various types of information that can be retrieved are described above in connection with the description of the SLGetInformation method. The licensing service then receives the call (504). The service retrieves the requested information from the license file that contains the bound right (506). The licensing service then places this information in a buffer (508), and returns to the calling application (510). The calling application then reads the contents of the buffer, and performs whatever actions it deems necessary based on the retrieved information.

[0135] It should be noted that the licensing service may not be aware of the meaning of the information that it is handling as part of an SLGetInformation call. As discussed above, the licensing framework provides a mechanism whereby a software vendor can create rights, and can associate information with the rights. The invention is not limited either to any particular type of information that can be associated with the right. When the information is retrieved from the

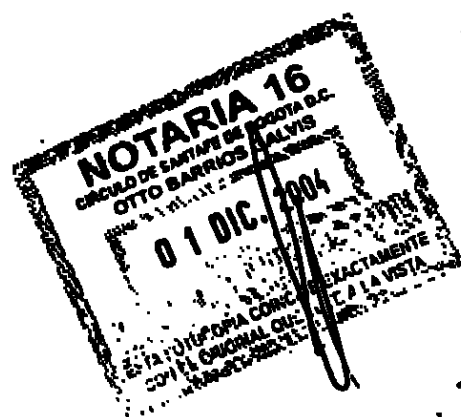


124
123

125

license, it is simply passed by the licensing service to the application in a buffer. The application then interprets the retrieved information, decides what actions to be taken based on that information, and uses its own security features to enforce the application's decision. (E.g., if, based on the retrieved information, the application decides to disable the print feature, the application contains the code that actually disables this features and, possibly, code that prevents a hacker from tampering with the disabling of the print feature.)

[0136] It is noted that the foregoing examples have been provided merely for the purpose of explanation and are in no way to be construed as limiting of the present invention. While the invention has been described with reference to various embodiments, it is understood that the words which have been used herein are words of description and illustration, rather than words of limitations. Further, although the invention has been described herein with reference to particular means, materials and embodiments, the invention is not intended to be limited to the particulars disclosed herein; rather, the invention extends to all functionally equivalent structures, methods and uses, such as are within the scope of the appended claims. Those skilled in the art, having the benefit of the teachings of this specification, may effect numerous modifications thereto and changes may be made without departing from the scope and spirit of the invention in its aspects.



What is Claimed:

1. A system for supporting the enforcement of a license for a computer program, the system comprising:

a licensing component that maintains a license store in which the license is stored, the license comprising a right in the software and a set of data associated with said right, the licensing component exposing a callable interface to the computer program, said callable interface comprising:

a right-consumption method which receives an identifier of said right from the computer program and determines whether the right can be exercised; and

an information-retrieval method which receives an identifier of said right from the computer program and provides said set of data, or information based on said set of data, to the computer program.

2. The system of claim 1, wherein said licensing component is usable by a plurality of computer programs, the computer program being included among said plurality of computer programs, wherein said callable interface further comprises:

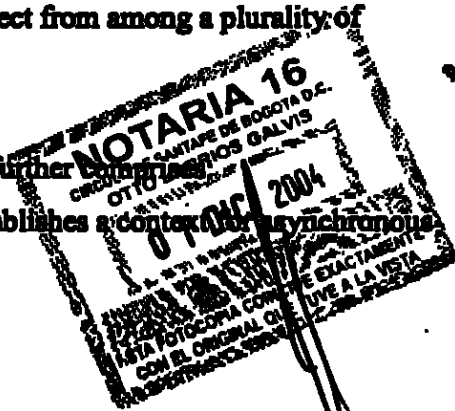
a handle-opening method that provides a handle to the computer program; wherein the rights-consumption method receives the handle from the computer program and uses the handle to identify the computer program from which a call to the rights-consumption method is received.

3. The system of claim 1, wherein the license is one of a plurality of licenses that are stored in said license store, and wherein the rights-consumption method causes the licensing component to select the license based on one or more factors comprising:

whether the license store is associated with the computer program; and

a conflict rule that determines which license to select from among a plurality of licenses that are associated with the computer program.

4. The system of claim 1, wherein said callable interface further comprises:
an asynchronous-context-initiator method that establishes a context; and
an asynchronous-context-terminator method that terminates the context.



processing and provides an identifier of said context to the computer program;
wherein said rights-consumption method receives the identifier of said context from said computer program and processes a right-consumption request asynchronously in response to receipt of the identifier of said context.

5. The system of claim 1, wherein the rights-consumption method determines whether the right can be exercised based on whether the right is identified in the license.

6. The system of claim 1, wherein the computer program and the licensing component execute on a machine, and wherein the rights-consumption method determines whether the right can be exercised based on whether the license is bound to said machine.

7. The system of claim 1, wherein the computer program is associated with a product identifier, and wherein the rights-consumption method determines whether the right can be exercised based on whether the license is bound to said machine or to a class of machines of which said machine is a member.

8. A method of restricting the use of a computer program associated with a license, the license specifying a right in the computer program, the method comprising:

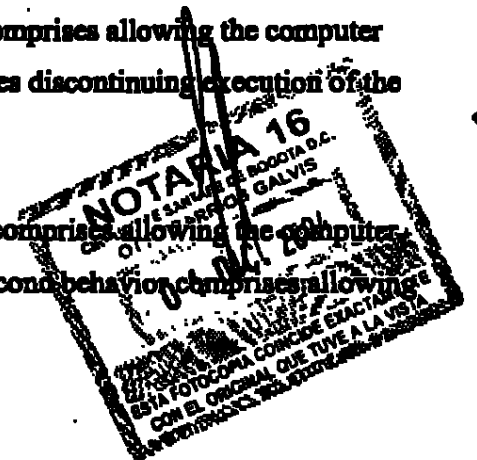
invoking a licensing service by making a first call to a first method of an interface of said licensing service, said first call being parameterized by an identifier associated with said right;

in response to said first call receiving an indication as to whether the right is exercisable; and

engaging in either a first behavior or a second behavior according to the indication.

9. The method of claim 8, wherein said first behavior comprises allowing the computer program to execute, and wherein said second behavior comprises discontinuing execution of the computer program.

10. The method of claim 8, wherein said first behavior comprises allowing the computer program to perform a first set of functions, and wherein said second behavior comprises allowing



the computer program to perform a second set of functions that is non-identical to said first set of functions.

11. The method of claim 8, wherein the right is associated with a set of data, wherein the method further comprises:

making a second call to a second method of said interface, said second method being parameterized by an indication of the right; and
in response to said second call, receiving said set of data.

12. The method of claim 11, further comprising:

directing the operation of the computer program based on said set of data.

13. The method of claim 8, further comprising:

making a second call to a second method of said interface; and
in response to said second call, receiving a handle;
wherein said second call is made prior to said first call, and wherein said first call is further parameterized by said handle.

14. The method of claim 8, further comprising:

making a second call to a second method of said interface; and
in response to said second call, receiving an asynchronous context;
wherein said second call is made prior to said first call, wherein said first call is further parameterized by said asynchronous context, and wherein the computer program performs at least one action while the first call is handled asynchronously.

15. The method of claim 8, wherein said first method determines whether the right is exercisable based on one or more factors comprising:

whether the license is bound to a machine or environment on which the computer program is executing;

whether the license or right is bound to a product identifier associated with the



whether the license or right has expired; and

whether the right has been consumed a number of times in excess of a right specified

16. A computer-readable medium having encoded thereon computer-executable instructions

receiving a first method call from the computer program, the first method call

determining that the right is contained in the license and is exercisable; and

returning to the computer program an indication that the right is exercisable.

17. The computer-readable medium of claim 16, wherein the indication comprises a binding

18. The computer-readable medium of claim 16, wherein said determining act is based on

19. The computer-readable medium of claim 16, wherein said determining act is based on

20. The computer-readable medium of claim 16, wherein said determining act is based on

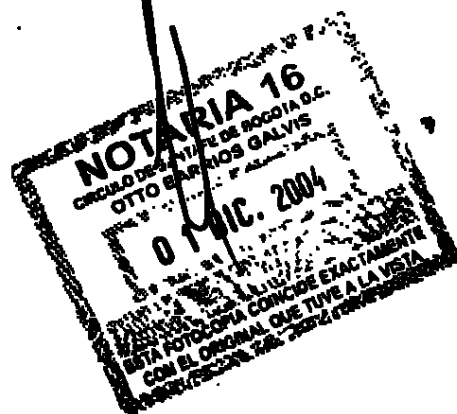
21. The computer-readable medium of claim 16, wherein said determining act is based on

22. The computer-readable medium of claim 16, wherein said determining act is based on

23. The computer-readable medium of claim 16, wherein the method further comprises:
receiving a second method call from the computer program; and
in response to the second method call, returning a handle to the computer program
that identifies the computer program;
wherein said first method call is performed subsequent to said second method call, and wherein said
first method call further identifies said handle.

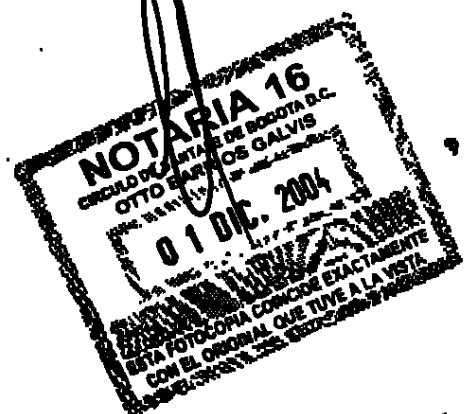
24. The computer-readable medium of claim 16, wherein the method further comprises:
receiving a second method call from the computer program;
in response to the second method call, returning an asynchronous context to the
computer program, wherein the first method call is executed subsequent to the second method call
and identifies said asynchronous context; and
executing the first method call asynchronously while the computer program performs
an action.

25. The computer-readable medium of claim 16, wherein the right is associated with a set of
data, and wherein the method further comprises:
receiving a second method call which indicates the right; and
in response to said second method call, providing the set of data to the computer
program.



ABSTRACT OF THE DISCLOSURE

A software licensing Application Programming Interface (API) that allows software products to use the license management functionality of a common service. A license specifies rights in a software product. The software product calls a consume method on the API in order to consume a right. If the right exists, the service binds the right to the license in which the right is found. The software product enforces the terms of the license by granting, or denying, access to some or all features depending on whether a valid instance of the right is found. Arbitrary data can be associated with a right. The API includes a method to retrieve data from a right that has been previously bound by the consume method.



131

PATENT APPLICATION

DECLARATION AND POWER OF ATTORNEY

ATTORNEY DOCKET NO. MSFT-3222

MS DOCKET NO. 305442.1

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below next to my name:

I believe I am (the original, first and sole inventor (if only one name is listed below) : an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled: **PROGRAMMING INTERFACE FOR LICENSING**

the specification of which is filed herewith unless the following box is checked:

- () was filed on _____ as US Application Serial No. or PCT International Application
Number _____ and was amended on _____ (if applicable).

I hereby state that I have reviewed and understood the contents of the above-identified specification, including the claims, as amended by any amendment(s) referred to above. I acknowledge the duty to disclose all information which is material to patentability as defined in 37 CFR 1.56.

Foreign Application(s) and/or Claim of Foreign Priority

I hereby claim foreign priority benefits under Title 38, United States Code Section 119 of any foreign application(s) for patent or inventor(s) certificate listed below and have also identified below any foreign application for patent or inventor(s) certificate having a filing date before that of the application on which priority is claimed:

COUNTRY	APPLICATION NUMBER	DATE FILED	PRIORITY CLAIMED UNDER 38 U.S.C. 119
			YES NO
			YES NO

POWER OF ATTORNEY:

As a named inventor, I hereby appoint the following attorney(s) and/or agent(s) associated with:

23377

and

David Berney Eppensner Reg. No. 35,499	Martin L. Shively Reg. No. 33,553	Ronald O. Zink Reg. No. 35,744	Patricia R. Barnes Reg. No. 37,038
Stacy Quinn Reg. No. 33,760	Jeffrey L. Rancik Reg. No. 34,590	Michael W. Bocimowski Reg. No. 28,692	John Wersah Reg. No. 32,322

to prosecute this application and transact all business in the Patent and Trademark Office connected therewith.

Send Correspondence to:	Direct Telephone Calls To:
Contact Name: Steven J. Reed Firm Name: Woodcock Washburn, LLP Firm Address: One Liberty Place, 46th Floor City, State and Zip: Philadelphia, PA 19103	Contact Name: Peter M. Ullman Contact Phone Number: (215) 564-8296



132

DECLARATION AND POWER OF ATTORNEY

ATTORNEY DOCKET NO. MSFT-2322

MS DOCKET NO. 303442.1

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Full Name of Inventor: Carla GauridiCitizenship: TurkeyResidence: 460 23rd Lane NEPost Office Address: Sammamish, Washington 98074

Inventor's Signature

10/24/2003

Date

Full Name of Inventor: Yuan-Pin Scott HsuCitizenship: TaiwanResidence: 18114 NE 25th StreetPost Office Address: Redmond, Washington 98072

Inventor's Signature

10/24/2003

Date

Full Name of Inventor: Ning ZhangCitizenship: United States of AmericaResidence: 20000 SE 9th StreetPost Office Address: Sammamish, Washington 98074

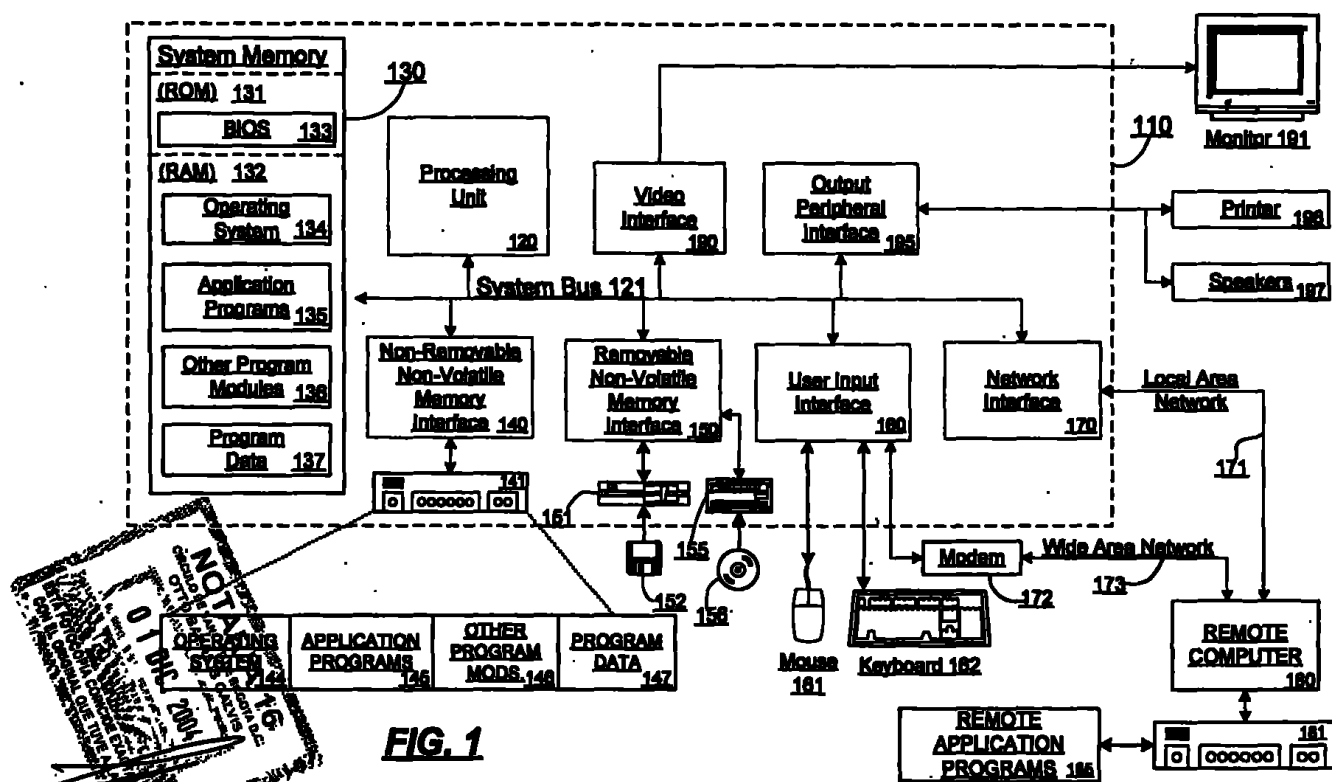
Inventor's Signature

10/24/2003

Date



Computing Environment 100



135
134

2/5

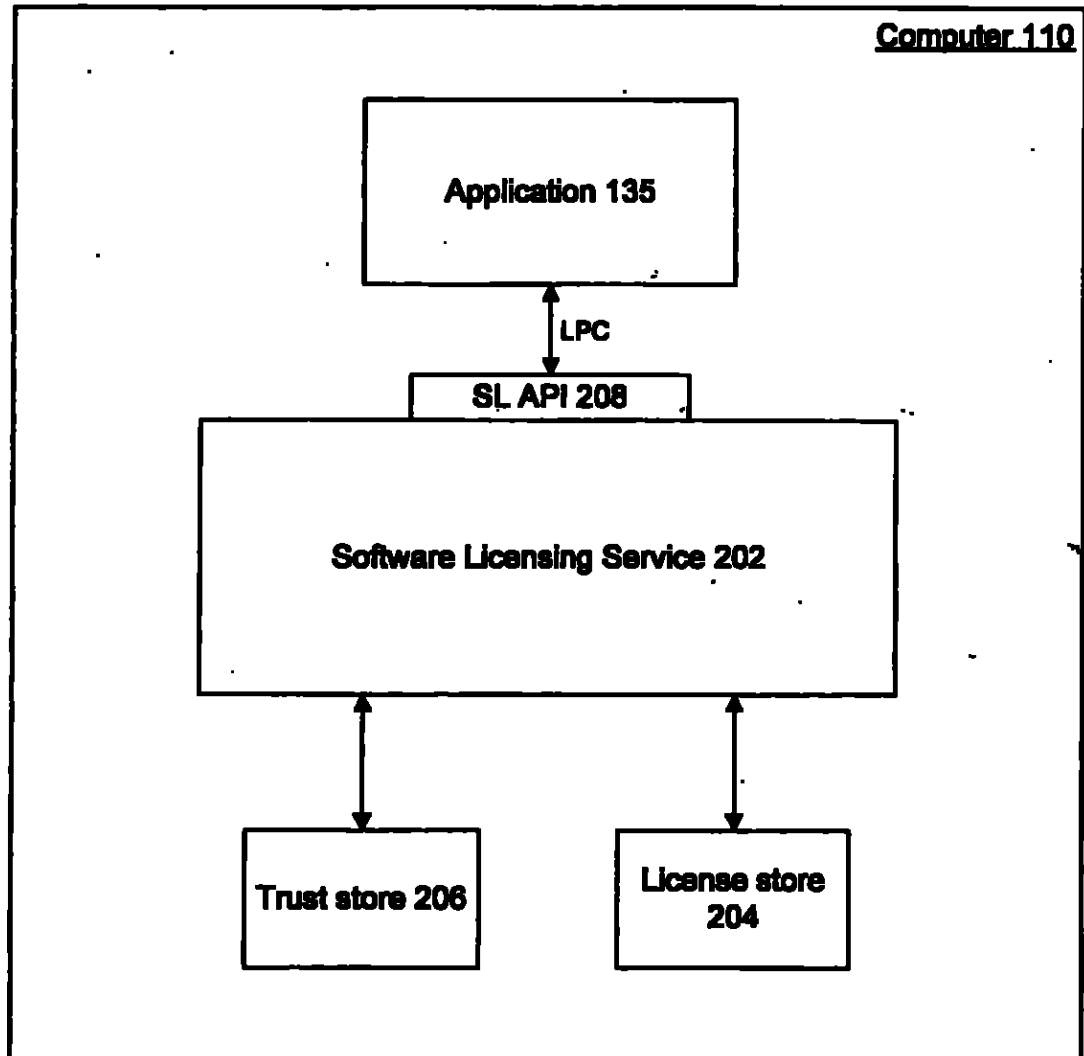


FIG. 2



135

3/5

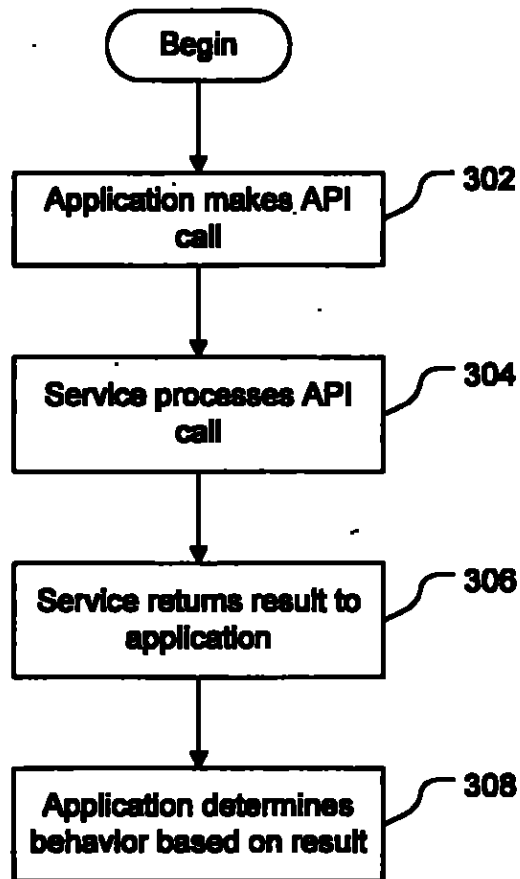
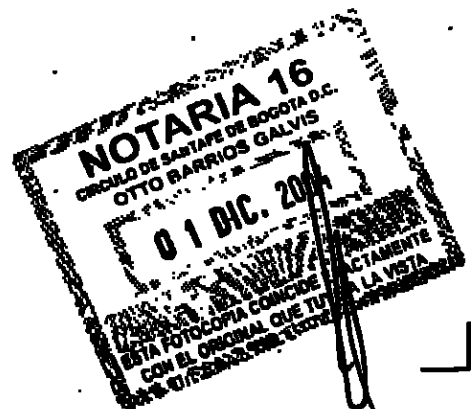


FIG. 3



57

137
136

4/5

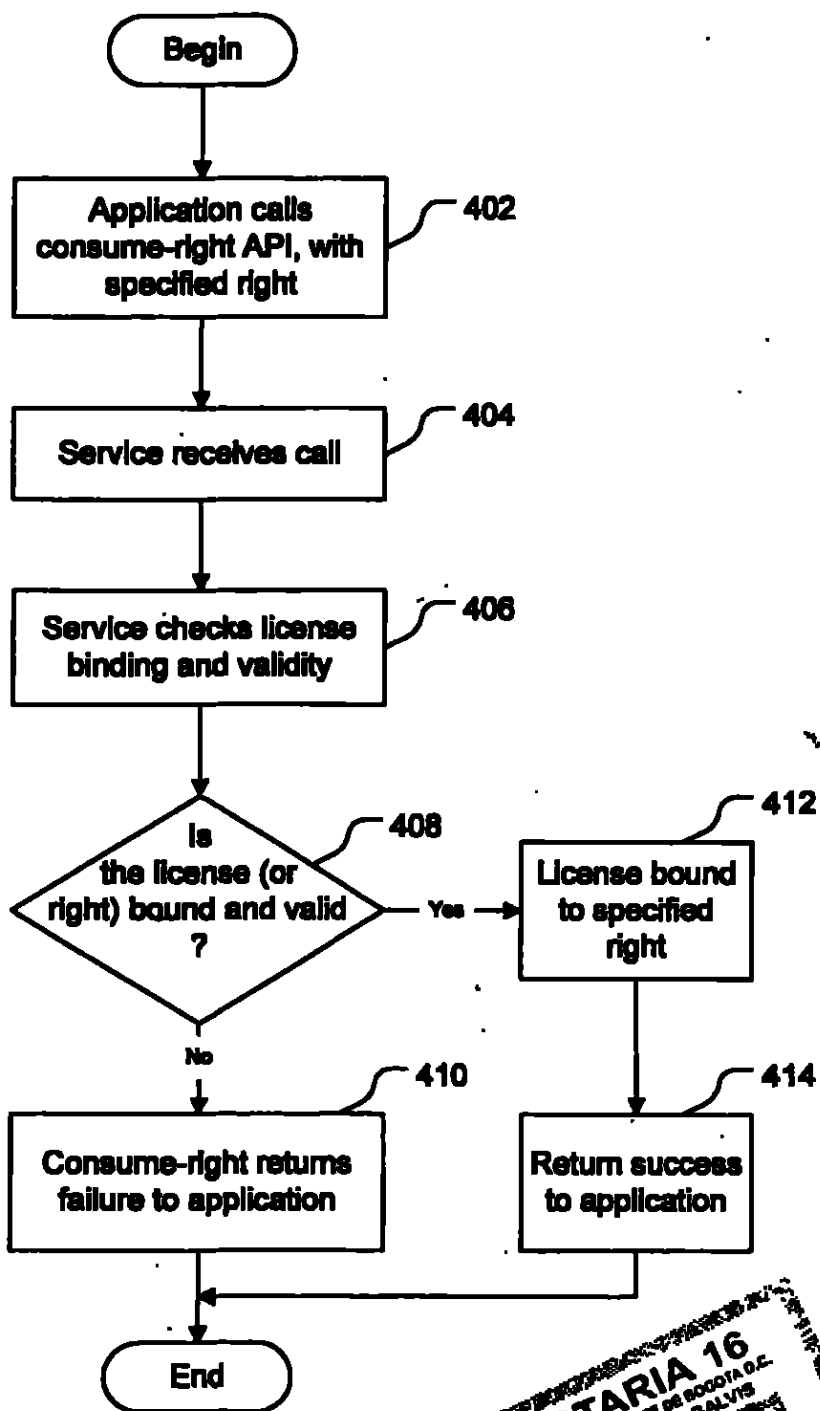
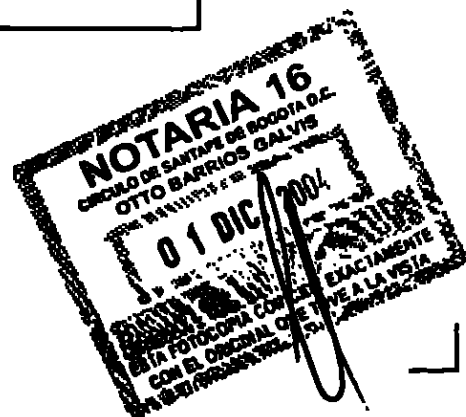


FIG. 4



158
137

5/5

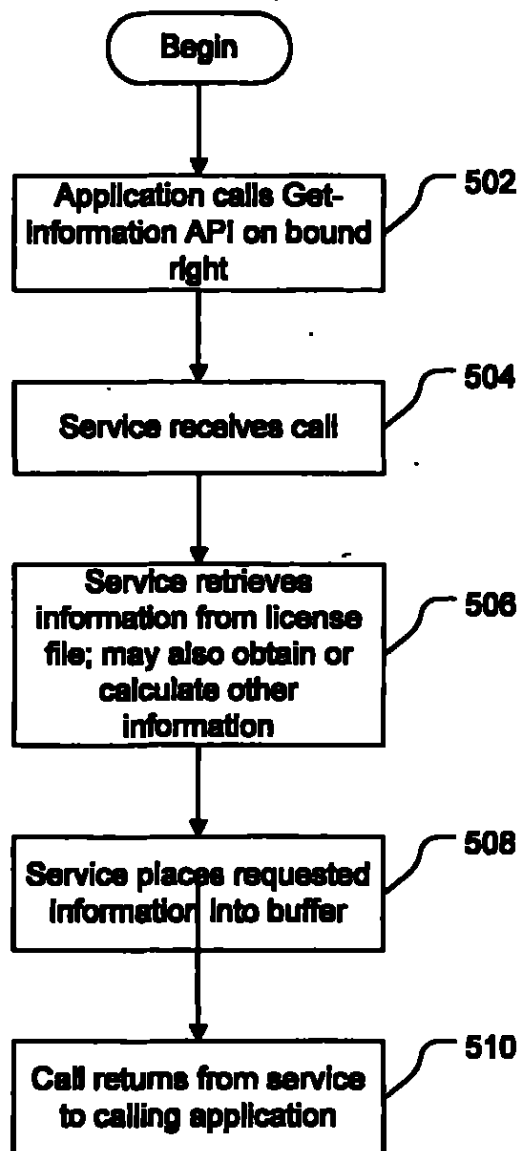


FIG. 5



138

ESTADOS UNIDOS DE AMERICA

A QUIEN PUEDA INTERESAR:

DEPARTAMENTO DE COMERCIO DE LOS ESTADOS UNIDOS

Oficina de Marcas y Patentes de los Estados Unidos

Julio 30 de 2004

CERTIFICAMOS POR LA PRESENTE QUE LO QUE SE ANEXA A ESTE CERTIFICADO ES UNA COPIA FIEL DE LOS REGISTROS DE LA OFICINA DE MARCAS Y PATENTES DE LOS ESTADOS UNIDOS, DE LOS DOCUMENTOS DE LA SOLICITUD DE PATENTE IDENTIFICADA A CONTINUACION, LA CUAL CUMPLIO LOS REQUISITOS PARA QUE LE FUERA OTORGADA UNA FECHA DE REGISTRO, DE CONFORMIDAD CON EL ARTICULO 111 DE LA LEY 35 DEL DEL CÓDIGO DE COMERCIO DE LOS ESTADOS UNIDOS (USC).

NÚMERO DE SOLICITUD: 10/692.868

FECHA DE REGISTRO: Octubre 24, 2003

Con la autoridad del

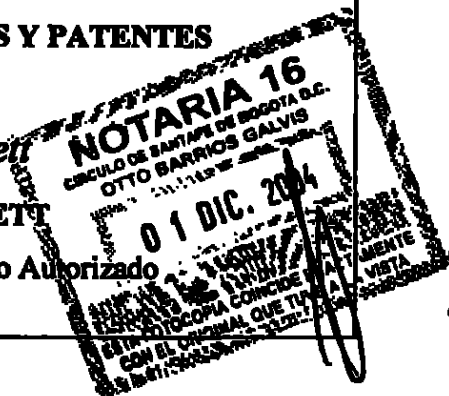
COMISIONADO DE MARCAS Y PATENTES

(Sello)

E. Bornett

E. BORNETT

Funcionario Autorizado



Victor Eugenio Cruz de Romerez
INTERPRETE OFICIAL
RES. 381 - 220479

46
139

FICHA No. MSFT-2822/305442.1

PATENTE

EN LA OFICINA DE MARCAS Y PATENTES DE LOS ESTADOS UNIDOS

En Referencia a la Solicitud de:

Caglar Gunyakti, Wen-Pin Scott Hsu, Ning Zhang

Para: INTERFACE DE PROGRAMACION PARA EL LICENCIAMIENTO

ETIQUETA DE CORREO INMEDIATO No: EL 970382875 US

FECHA DE DEPOSITO: Octubre 24 de 2003

EL 970382875 US

☒ Solicitud de Patente por correo ☐ Diseño por correo ☐ Secuencia por correo

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Señores

CARTA DE ENVÍO DE UNA SOLICITUD DE PATENTE

Para registro, con la presente enviamos:

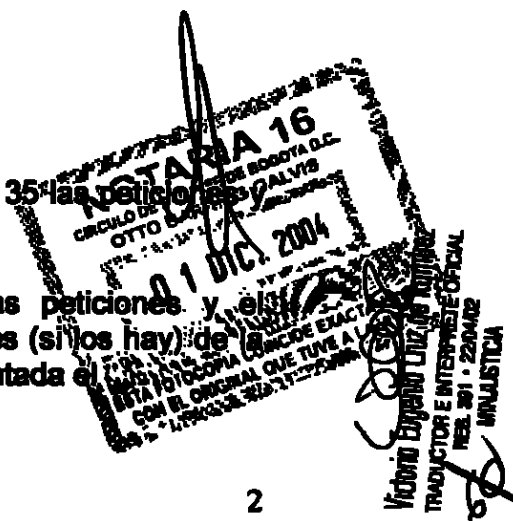
X Solicitud de Utilidad de conformidad con el inciso 1.53 (b) del artículo 37 del Código de Regulaciones Federales (Code of Federal Regulations)

La Solicitud anteriormente identificada incluye lo siguiente:

☐ Una Solicitud de No publicar

X Una Especificación que incluye en las páginas 1 a 35 las peticiones y el resumen.

☐ Una copia de la especificación, que incluye las peticiones y el resumen (páginas del al), los dibujos y los apéndices (si los hay) de la Solicitud, de fecha anterior, con el Número y presentada en



141
140

a la cual no se le ha añadido nada. Por medio de la presente, dicha Solicitud de fecha anterior queda incorporada a la presente solicitud de la referencia.

☐ Una nueva Declaración o Juramentación y Poder de Representación Legal sin formalizar.

X Una Nueva Declaración o la Juramentación y el Poder de Representación Legal Formalizados.

☐ Se adjunta una copia de la Juramentación o la declaración, formalizadas, registrada con la solicitud de fecha anterior, identificada anteriormente.

☐ Un Poder de representación legal Asociado formalizado por quien figura en la copia de la Declaración y el Poder de Representación Legal registrados con la presente.

☐ Una copia del Poder de representación Legal con Revocación, registrada en la Solicitud No. , de fecha anterior.

☐ Una Declaración Firmada en la cual se borra al (los) inventor (es) nombrado (s) en una solicitud anterior.

☐ La presente solicitud reclama prioridad extranjera, de acuerdo con el artículo 119 de la Ley 35 del USC, para la solicitud No. , registrada el , en (País).

☐ Una Copia Certificada de cada una de las solicitudes anteriores para las cuales se reclama prioridad:

☐ se adjunta.

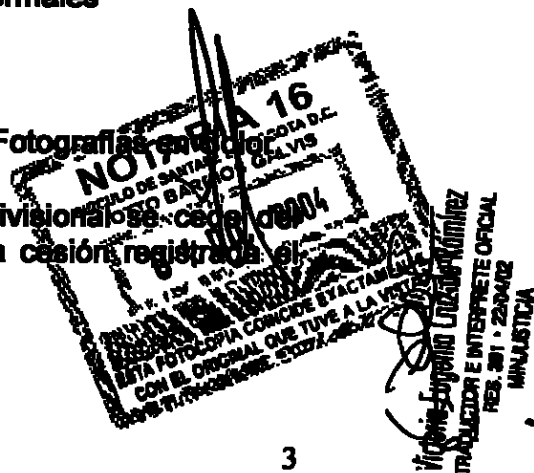
☐ Fue registrada con la anterior Solicitud No. , registrada el .

X Cinco (5) Páginas de Dibujos y / o Fotografías Formales

X Debe publicarse la Figura 3

☐ Se adjunta la Petición de Aceptación de Fotografías en color

☐ La presente solicitud de continuación y / o divisional se registra a registro a , tal como se pone en evidencia en la cesión registrada en el Carrete No. , Cuadro No.



☒ Una nueva Cesión Formalizada que le transfiere los derechos a Microsoft Corporation.

☒ Una cubierta con el Formato de Inscripción.

☒ La Tarifa de Inscripción: US\$40,00

☐ Se adjunta una Modificación Preliminar.

☐ Un Listado de Secuencia, que consiste de las páginas 1 a

☐ Se adjunta un diskette de computador que contiene un Listado de Secuencia.

☐ El formato, legible por computador, que acompaña la solicitud anteriormente identificada es idéntico al registrado con la Solicitud No. , registrada el . De conformidad con el artículo 182(e) de la ley 37 del Código de Regulaciones Federales - CFR, sírvanse utilizar el listado de secuencia registrado el , recibido por la Oficina de Marcas y Patentes (PTO) el , como formato legible por computador registrado en esa solicitud como formato legible por computador para la presente solicitud. Se entiende que Oficina de Marcas y Patentes le hará al formato legible por computador que será usado para la presente solicitud los cambios necesarios en la Número de la Solicitud y en su fecha de registro.

☐ Una copia en papel del Listado de Secuencia:

☐ Se registró con la Primera solicitud el anteriormente identificada, la cual fue recibida por la Oficina de Marcas y Patentes el

☐ Se adjunta a la presente, en una Modificación Preliminar, para ser incorporada dentro de la especificación.

☐ Se adjunta una Declaración de Apoyo a la Presentación de Información de Secuencia.

☐ La Declaración de Divulgación de información que incluye:

☐ El Formato 1449

☐ Las Copias de las Referencias enumeran en el Formato PTO-1449 adjunto.

☐ Una copia de la Petición de Ampliación de Tiempo (Petition for Extension of Time), tal como se registró en el caso anterior de copendencia.



Victorio Espinoza Ortiz de Ramiñez
TRADUCTOR E INTERPRETE OFICIAL
FECH. 201 - 220402
MINISTERIO DE JUSTICIA

143
142

☐ Material Que Acompaña, tal como se describe:

X Tarjeta Postal para Devolución del Recibo (debe venir detallada y especificada).



[Signature]
Ricardo Espinosa Cruz de Ramírez
TRADUCTOR E INTERPRETE OFICIAL
REG. 391 - 220402
MINISTERIO DE JUSTICIA

PATENTE

☐ El (los) Solicitante(s), por de sus apoderados, reclaman la categoría de entidad pequeña de conformidad con el artículo 1.27 de la ley 37 del CFR (Code of Federal Regulations) como . (ESCOJA UNO: Inventor Independiente, Pequeña Empresa Comercial, u Organización Sin Animo de Lucro).

X Se adjuntan dos Cheques: (1) por la suma de **US\$880,00 (Tarifa de radicación de la Solicitud)**; y (2) por la suma de **US\$40,00 (Tarifa de Radicación de la Cesión)**. Por favor cárguenos cualquier faltante o acreditemos cualquier sobrepago a la Cuenta Corriente No.23-3050.

X Por medio de la presente, solicitamos al Comisionado que nos otorgue una extensión temporal durante el lapso de tiempo adecuado en caso de ser necesaria, en lo que atañe a esta radicación o cualquier otra radicación que



Victorino Eugenio Cruz de la Cruz
 TRADUCTOR E INTERPRETE OFICIAL
 REG. 301 • 220442Z
 MIN. JUSTICIA

145
144

presente a la Oficina de Marcas y Patentes de los Estados Unidos referentes a la solicitud anteriormente identificada, durante el trámite de esta solicitud. El Comisionado queda autorizado, además, a hacer el cargo a la Cuenta Corriente No.23-3050 de cualquier tarifa relacionada con dicha extensión temporal. Esta página se presenta por duplicado.

X Dirija toda la correspondencia al Usuario No.23377.

23377

PATENT TRADEMARK OFFICE

X Dirija todas las llamadas telefónicas a:

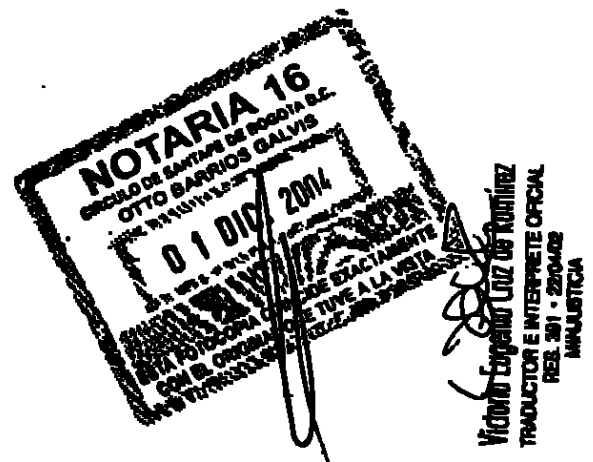
Peter M Ullman
Woodcock washburn LLP
One Liberty Place – 48th Floor
Philadelphia, PA 19103
Teléfono No: (215) 568 3100
Fax No: (215) 568 3439

En caso de presentarse cualquier deficiencia con respecto a esta solicitud, incluyendo deficiencias en el pago de las tarifas, falta de diligenciamiento en las partes de la solicitud o cualquier otra cosa, solicitamos respetuosamente a la Oficina de Marcas y Patentes de los Estados Unidos que notifique con prontitud al suscrito.

Fecha: Octubre 24 de 2003

(Firma)

Peter M. Ullman
Registro No.43.963



[Handwritten signature]

146
105

INTERFASE DE PROGRAMACION PARA EL LICENCIAMIENTO

CAMPO DE LA INVENCION

[0001] la actual invención se relaciona generalmente con el campo del software de computadora, y, más particularmente, con un interfase de programación que apoya la aplicación de licencias electrónicas.

ANTECEDENTE DE LA INVENCION

[0002] El software producidos en el comercio de la tradicionalmente se ha hecho disponible bajo una licencia que define los términos permitidos del uso del software. Cuando la práctica del software licenciado comenzó, la licencia tomó generalmente la forma de un documento jurídico que definió los derechos del usuario con respecto al software. Tal documento confió en el sistema legislativo para la aplicación. Esto se ha vuelto deseable para la aplicación de las licencias electrónicamente, es decir, es deseable que un programa de computadora contenga el código que desalienta activamente o previene el uso del software de una manera que sea contraria a la licencia.

[0003] La mayoría de software que preve la aplicación electrónica de la licencia proporciona su propia infraestructura para manejar el licenciamiento del software y el uso de las licencias. Así, un producto de software comercial típico puede incluir no solamente el código para realizar la función de la base del producto, pero puede también llevar con él el código para obtener, evaluar, proteger, y para manejar las licencias del software. Para que cada vendedor del software desarrolle e incorpore esta infraestructura en su software es a menudo una duplicación derrochadora de esfuerzo. Es por lo tanto deseable proporcionar un sistema que realice las funciones básicas relacionadas con el software que licencia, donde el sistema se puede utilizar por una amplia variedad de usos de software en una uniforme y definida manera.

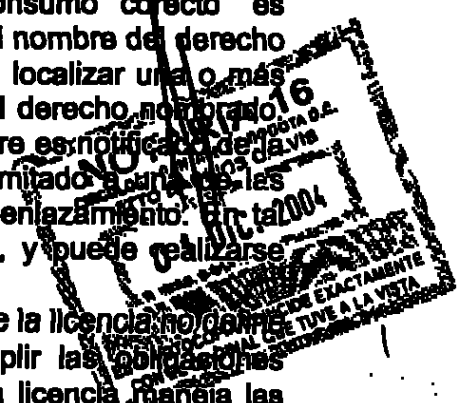
[0004] En la vista precedente, hay una necesidad de un mecanismo que supera las desventajas del arte anterior.

EL RESUMEN DE LA INVENCION

[0005] La actual invención proporciona una licencia de software de la Interfase del Programa de Aplicación (API) que proporcione ciertas funciones de licencia para el uso de productos de software. Un servicio de licencia realiza funciones referentes al uso de licencias, y expone estas funciones a los productos de software con el API. El servicio realiza funciones tales como la obtención de las licencias, almacenamiento y el manejo, protegiendo la licencia de tratar de forzar, de evaluar la validez de una licencia, y de evaluar si una licencia está limitada correctamente al producto de la máquina y/o de software en el cual se utiliza. El software puede hacer uso esta funcionalidad llamando los métodos del API.

[0006] En un uso típico del API, un producto de software llama un método "abre" del API para obtener un manejo único que sea utilizado por el servicio de la licencia para identificar la aplicación. El producto de software entonces llama al método API "consumo corecto". "consumo," en este contexto, significa el ejercicio de un derecho especificada. La llamada al método "consumo corecto" es paremitrizada por el manejo del producto de software, y por el nombre del derecho a ser consumido. El servicio de la licencia entonces procura localizar una o más licencias válidas, correctamente nombradas que contienen el derecho nombrado. Si no existe ninguna licencia, entonces el producto de software es notificado de la falta. Si existen tales licencias, entonces el derecho está limitado a una de las licencias, y el producto de software llamado se notifica del enlazamiento. En tal caso, el producto de software sabe que existe el derecho, y puede realizarse cualesquiera funciones que se asocian a este derecho.

[0007] En una personalización preferida, el servicio de la licencia no permite lo que puede o no puede bajo el derecho, o haga cumplir las condiciones explícitos en el uso del software. Más allá, el servicio de la licencia maneja las licencias de una manera tal que un producto de software pueda determinarse



66

147
106

llamando el API si el derecho existe o no, de modo que el software pueda comportarse de acuerdo. Por ejemplo, un derecho se puede llamar "ejecución," indicando que el usuario tiene la derecho de ejecutar el producto de software. El producto de software puede utilizar el API para determinar si (correctamente enlazado, y no-expirado) hay un derecho válido para ejecutar el software. Sin embargo, si la llamada del API vuelve con una indicación de que no hay derecho para ejecutar el software, es decisión del software cesar la operación o tomar otra acción basada en la no existencia de este derecho.

[0008] Una derecho puede ser asociado con la información, que llega a estar disponible después de que una llamada satisfactoria a el método "consumo correcto". Por ejemplo, un producto de software dado puede tener reglas individuales alrededor de cuando es permitido editar, imprimir, grabar, etc., y estas reglas se pueden almacenar en la licencia que contiene el derecho. El API proporciona un método "obtener información" que permite recuperar esta información de la licencia.

[0009] Otras características de la invención se describen abajo.

BREVE DESCRIPCIÓN DE LOS GRAFICAS

[0010] El resumen precedente, así como la descripción detallada siguiente de personalizaciones preferidas, se entiende mejor cuando está leída conjuntamente con los dibujos adjuntos. Con el fin de ilustrar la invención, se muestra en las construcciones ejemplares de los dibujos de la invención; sin embargo, la invención no se limita a los métodos y a los instrumentoss específicos divulgados. En los dibujos:

[0011] La fig. 1 es un diagrama de bloque de un ambiente de cómputo del ejemplo en el cual los aspectos de la invención puedan ser puestos en ejecución;

[0012] La fig. 2 es un diagrama de bloque de una arquitectura en la cual un sistema realiza funciones de licenciamiento y expone un API para el uso por los productos de software;

[0013] La fig. 3 es un organigrama de un método con el cual un producto de software utiliza un API de licenciamiento;

[0014] La fig. 4 es un organigrama de un método por el cual un producto de software consume un derecho; y

[0015] La fig. 5 es un organigrama de un método por el cual un producto de software recupera la información referente a un derecho consumido.

DESCRIPCION DETALLADA DE LA INVENCION

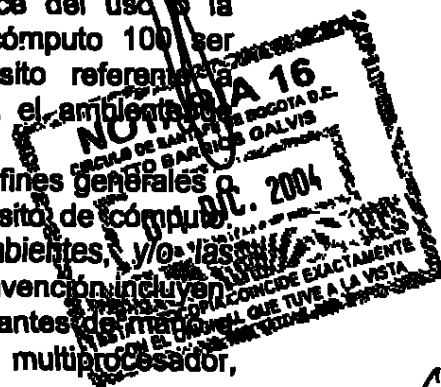
Overview

[0016] el uso del software comercial es gobernado típicamente por una licencia, y ha llegado a ser cada vez más común que esta licencia sea incorporada a una forma electrónica que se puede hacer cumplir por el software sí mismo. Un desafío de crear un sistema de licenciameinto electrónico es que una infraestructura es necesaria para manejar el uso de las licencias. Replicar esta infraestructura para cada producto de software es incómodo y derrochador. La actual invención proporciona un API que permite que diversos productos de software utilicen una infraestructura común que realiza varias funciones de licenciamiento.

ARREGLO DE COMPUTO EJEMPLAR

[0017] Fig. 1 muestran un ambiente de cómputo ejemplar en el cual los aspectos de la invención pueden ser implementados. El ambiente del sistema de cómputo 100 es solamente un ejemplo de un ambiente de cómputa conveniente y no se piensa sugerir ninguna limitación en cuanto al alcance del uso o la funcionalidad de la invención. No debería el ambiente de cómputo 100 ser interpretado como teniendo cualquier dependencia o requisito referencial a cualquiera o a la combinación de componentes ilustrados en el ambiente de funcionamiento ejemplar 100.

[0018] La invención es operacional con numeroso otros fines generales o ambientes o configuraciones especiales del sistema del propósito de cómputo. Ejemplos de los sistemas de cómputo bien conocidos, ambientes, y/o las configuraciones que pueden ser conveniente para el uso con la invención incluyen pero no se limitan a, ordenadores personales, servidores, ayudantes de escritorio, los dispositivos de computadora portátil, sistemas basados en multiprocesador,



62

व४
147



•

El teléfono debe tener una o más de sus líneas de información en la comunicación incluyen una directa cableada y otras telefónicas. Los datos deben también.

68

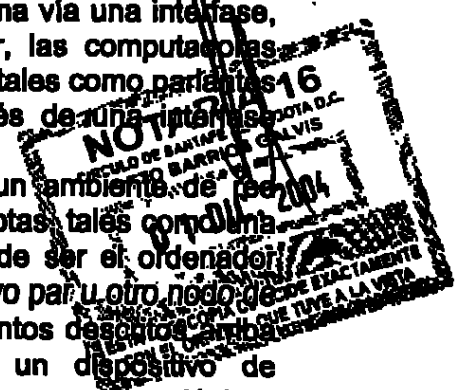
145
148

[0022] La memoria de sistema 130 incluye medios de almacenamiento en forma de memoria volátil y/o permanente tal como memoria sólo para leer (ROM) 131 y memoria de acceso aleatorio (RAM) 132. Un sistema básico 133 (BIOS) de entrada-salida, conteniendo las rutinas básicas que ayudan a transferir la información entre los elementos dentro de la computadora 110, por ejemplo durante el inicio (start-up), se almacena típicamente en ROM 131. La RAM 132 contiene típicamente los módulos de datos y/o del programa a los cuales sea inmediatamente accesible y/o actualmente siendo encendido por la unidad de proceso 120. A modo de ejemplo, y sin limitación, Fig. 1 ilustra el sistema operativo 134, los programas de aplicación 135, otros módulos del programa 136, y los datos del programa 137.

[0023] La computadora 110 puede también incluir otro medio de almacenamiento removible/no-removible, volátil/no volátil. A modo de ejemplo solamente, la Fig. 1 ilustra una unidad de disco duro 140 que lee desde o escribe en medios no removibles, permanentes, una unidad de disco magnético 151 que lee o escribe en un disco magnético removible, permanente 152, y a una unidad de disco óptico 155 que lee desde o escribe en un disco óptico removible, permanente 156, tal como una ROM de CD u otros medios ópticos. Otros medios de memoria removible/no-removible, volátil/no volátil, que se pueden utilizar en el ambiente de funcionamiento ejemplar incluyen, pero no se limitan a, los casetes de cinta magnética, las tarjetas de memoria flash, los discos versátiles digitales (DVDs), cinta video digital, RAM de estado sólido, ROM de estado sólido, y similares. La unidad de disco duro 141 está conectada típicamente con el bus 121 del sistema a través de una interfase fija de memoria tal como la interfase 140, y la unidad de disco magnético 151 y la unidad de disco óptico 155 son conectados típicamente con el bus 121 del sistema por una interfase removible de la memoria, tal como la interfase 150.

[0024] Las unidades y sus medios asociados de la memoria interna discutidos arriba e ilustrados en Fig. 1, proporcionan el almacenamiento de instrucciones legibles por computador, de estructuras de datos, de módulos de programa y de otros datos para la computadora 110. En la Fig. 1, por ejemplo, la unidad de disco duro 141 se ilustra como almacena el sistema operativo 144, los programas de aplicación 145, otros módulos 146 del programa, y los datos del programa 147. Observe que estos componentes pueden ser iguales o diferentes del sistema operativo 134, de los programas de aplicación 135, de otros módulos del programa 136, y de los datos del programa 137. El sistema operativo 144, los programas de aplicación 145, otros módulos del programa 146, y los datos del programa 147 son dados diversos números aquí para ilustrar que, en un mínimo, ellos son copias diferentes. Un usuario puede incorporar comandos y la información en los dispositivos de entrada directos de la computadora 20 tales como un teclado 162 y el dispositivo apuntador 161, designado comúnmente como ratón, un Trackball (Mouse de Esfera) o un touchpad (Mouse de cojín de Tacto). Otros dispositivos de entrada (no mostrados) pueden incluir un micrófono, una palanca de mando, un cojín de juego (gamepad), un plato de satélite, un digitalizador (scanner), o similares. Éstos y otros dispositivos de entrada están conectados a menudo con la unidad de proceso 120 a través de la interfase 160 de entrada del usuario que se junta al bus del sistema, pero se pueden conectar por otras estructuras de interfase de bus, tales como un puerto paralelo, un puerto de juego, o un bus serial universal (USB). Un tipo de monitor 191 u otro dispositivo de pantalla también está conectado con el bus 121 del sistema vía una interfase, tal como una interfase de video 190. Además del monitor, las computadoras pueden también incluir otros dispositivos de salida periféricos tales como parlantes 197 e impresora 196, que pueden ser conectados a través de una interfase periférica 195 de salida.

[0025] La computadora 110 puede funcionar en un ambiente de red usando conexiones lógicas a unas o más computadoras remotas, tales como una computadora remota 180. La computadora remota 180 puede ser el ordenador personal, el servidor, el enrutador, la PC de la red, el dispositivo par u otro nodo de red común, e incluye típicamente muchos o todos los elementos descritos arriba relacionados a la computadora 110, aunque solamente un dispositivo de almacenamiento de memoria 181 se ha ilustrado en Fig. 1. Las conexiones lógicas



2.9

representadas en la Fig. 1 incluyen una red de área local (LAN) 171 y una red de área amplia (WAN) 173, pero pueden también incluir otras redes. Tales ambientes de red son comunes en oficinas, redes de ordenadores de empresas grandes, intranets e Internet.

[0026] Cuando está utilizada en un ambiente de red LAN, la computadora 110 está conectada con la LAN 171 a través de una interfase o de un adaptador de red 170. Cuando se utiliza en un ambiente de red WAN, la computadora 110 incluye típicamente un módem 172 u otro para establecer comunicaciones sobre la WAN 173, tal como Internet. El módem 172, que puede ser interno o externo, se puede conectar con el bus 121 del sistema vía la interfase 160 de entrada de usuario, u otro mecanismo apropiado. En un ambiente interconectado, los módulos de programa representados relacionados con la computadora 110, o partes de ella, se pueden almacenar en el dispositivo de almacenamiento remoto de la memoria. A modo de ejemplo, y sin limitación, la Fig. 1 ilustra los programas de aplicación remota 185 como residente en el dispositivo de memoria 181. Es importante mostrar que las conexiones de red son ejemplos y que pueden ser utilizados otros medios de establecer un puente de comunicaciones entre las computadoras.

Servicio de Licenciamiento de Software

[0027] La Fig. 2 muestra un sistema de ejemplo que proporciona un servicio que licenciamiento del software 202. El servicio de licenciamiento de software 202 funciona dentro de la computadora 110 (mostrada en la Fig. 1). En un ejemplo, el servicio de licenciamiento de software 202 es parte de un sistema operativo que se ejecuta en la computadora 110. El servicio de licenciamiento del software mantiene un almacén de la licencia 204 en el cual los archivos de la licencia para el software se almacenen. Los archivos de la licencia pueden, por ejemplo, ser los archivos de derechos del lenguaje enriquecido (XrML – eXtensible Rights Markup Language) que especifican los derechos al software, y que también pueden especificar varios tipos de condiciones en el ejercicio de esos derechos. El software 204 también mantiene un almacén de confianza 206. El almacén de confianza 206 almacena datos dinámicos autenticables, de una manera de forzar-resistencia; el almacén de confianza 206 almacena los datos que se utilizan en el proceso de la validación de la licencia. Por ejemplo, ciertas licencias pueden tener fechas de vencimiento, y, para evitar que los datos de expiración sean evitados con la restauración no actualizada del reloj, el tiempo actual (y el tiempo transcurrido) se pueden almacenar periódicamente en el almacén de confianza 206 para asegurarse de que el reloj se está moviendo siempre hacia adelante.

[0028] El servicio de licenciamiento de software 202 maneja el almacén de licencia 204 y el almacén de confianza 206, y también realiza varias funciones relacionadas con licenciamiento de software. Por ejemplo, el servicio licenciamiento de software 202 puede contener los módulos que analizan los archivos de la licencia, los módulos que hacen cumplir la obligación de una licencia a una máquina en particular y/o a un caso particular de un producto de software, y un contador de tiempo (timer/counter) seguro (que usa el almacén de confianza 206 en la manera descrita arriba).

[0029] El servicio de licenciamiento de software 202 expone una interfase de programación de aplicación (API) 208 que permite al software de aplicación (tal como la aplicación 135) el servicio de licenciamiento de software 202. La aplicación 135 puede invocar las características del servicio de licenciamiento de software 208 haciendo la llamada local del procedimiento (LPC) a los métodos del API 208. Un sistema de ejemplo de los métodos del API que se puede exponer por el servicio de licenciamiento de software 202 se describe abajo.

[0030] La manera en qué el API 208 es utilizado por una aplicación se describe con referencia a la Fig. 3. Inicialmente, la aplicación hace una llamada del API (302). El servicio 202 procesa la llamada del API (304), y retorna el resultado de la llamada del API a la aplicación 306. Por ejemplo, una llamada del API puede solicitar ejercitar ("consume") un derecho concedida en una licencia, o para recuperar la información de una licencia. La aplicación entonces recibe el

resultado de la llamada del API, y se determina, basado en ese resultado, qué el comportamiento de la aplicación debe ser (308). Es decir el servicio de licenciamiento de software 202, en una personalización preferida, no se obliga una licencia directamente, sino que proporciona la infraestructura a través de la cual las licencias pueden ser manejadas y utilizadas. Por ejemplo, si una aplicación hace una llamada del API para consumir un derecho y el servicio 202 determina que no hay licencia válida que conceda este derecho, el servicio 202, en una personalización preferida, no evita que la aplicación funcione, sino que por el contrario informa a la aplicación que el derecho no está disponible. Así, la aplicación puede utilizar sus propios mecanismos para determinar qué hacer en respuesta a la indisponibilidad del derecho. Esta faceta del API proporciona a vendedores del software la flexibilidad de decidir cómo la infraestructura de licenciamiento proporcionada por el servicio 202 debe ser utilizada. En otra implementación, la aplicación se puede limitar a la funcionalidad del servicio de licenciamiento

Ejemplo de Licenciamiento de Software API

[0031] El siguiente ejemplo es un conjunto de métodos de API que se pueden exponer por un servicio de licenciamiento de software:

SLOpen

[0032] La función de SLOpen abre una SL manija de contexto del cliente que se debe utilizar para todas las llamadas subsecuentes del SL API. (a través de las descripciones del API del ejemplo, el "SL" referirá al servicio de licenciamiento de software. Así, la manija del contexto del cliente SL es la manija usada por un cliente para comunicarse con el servicio de licenciamiento de software.)

HRESULT

SLOpen(

CONST GUID * pguidApp,

HSLC * phSLC

);

Parameters

pguidApp

[0033] [Entrada] El apuntador para la aplicación GUID que identifica únicamente una aplicación. Si este argumento es NULO, se vuelve un error de E_INVALIDARG.

phSLC

[0034] [Salida] El manejo del contexto del cliente SL o INVALID_HANDLE_VALUE si falla.

Observaciones (Remarks)

[0035] Utilizan la función SLClose para cerrar una manija del contexto retornada por SLOpen.

[0036] La aplicación GUID: Identificación (ID) única de la aplicación. Para la versión de WINDOWS del conjunto de aplicación de MICROSOFT OFFICE, WinWord tiene una aplicación GUID la cual es diferente de la aplicación GUID de Excel. Para Windows, Windows en sí mismo es una aplicación, aunque esta compuesto de muchos programas.

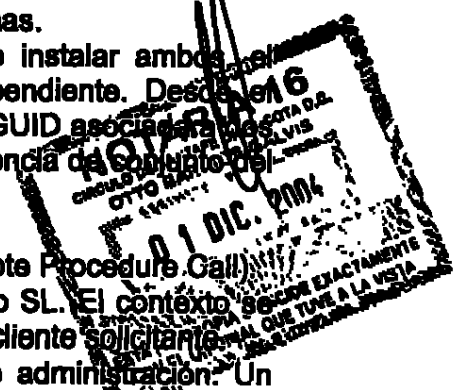
[0037] En el caso de la oficina, el usuario puede instalar ambos el conjunto de Office y WinWord como producto independiente. Desde el punto de vista SL, WinWord tiene la misma aplicación GUID asociada a los productos GUID. Es decir WinWord puede utilizar la licencia de conjunto del producto o la del producto de WinWord.

[0038] Cuando SLOpen tiene éxito:

[0039] Se ha establecido un atascamiento RPC (Remote Procedure Call)

[0040] Una memoria del contexto se crea en servicio SL. El contexto se utiliza para guardar el estado de la información para el cliente solicitante.

[0041] El manejo de SLC es como un archivo de administración. Un proceso puede abrir manijas múltiples de contexto del SL pero las manijas son válidas dentro del proceso del solicitante.



[Handwritten signature]

152
151

Returns

[0042] Éxito o falla

SLClose

[0043] La función SLClose cierra una manija abierta del contexto de cliente SL. Cualquier información en el contexto se libera automáticamente.

HRESULT

SLClose(

HSLC) *hSLC*

);

Parámetros

hSLC

[0044] [Entrada] La administración al cliente actual SL que maneja el contexto

Observaciones

[0045] Cuando se hace SLClose, el enlace del RPC ese liberado, y se destruye el contexto.

Returns (Retomo)

[0046] Éxito o falta.

SLInstall

[0047] La función de SLInstall instala las licencias de aplicación y registra la información de las aplicaciones.

HRESULT SLInstall(

HSLC *Hslc,*

(CONST SL_PRODKEY * *pAppPrdKey,*

DWORD *dwNumOfApps,*

CONST GUID * *pguidApps,*

DWORD *dwNumOfLicFiles*

PCWSTR *ppszLicFiles[]*

BOOL *bVerify*

);

Parameters

hSLC

[0048] [in - Entrada] Administración del actual manejo del contexto de cliente SL

pAppPrdKey

[0049] [in - Entrada] Estructura de clave del producto de la aplicación. La

152



153
152

clave del producto puede ser de formato clave del producto de Microsoft o del formato clave del producto de la aplicación.

```
typedef struct _tagSL_PRODKEY
{
    DWORD cbSize; //tamaño del SL_PRODKEY

    Structure //(Estructura)

    DWORD dw Version; //versión de la estructura
    SL_PRODKEY

    WCHAR szProdKey[MAX_PRODKEYSTR_SIZE+1];

    SL_PRODKEY_TYPE eProdKeyType; //tipo de clave del Producto

    SL_CUSTOM_PRODKEY_INFO CustomPrdKeyInfo; //tipo de clave del
    Producto

} SL_PRODKEY;
```

El eProdKeyType puede ser uno de valores siguientes:

SL_PRODKEY_CUSTOM

SL_PRODKEY_MS2002

SL_PRODKEY_MS2003

[0050] Si el tipo de clave de producto no es un producto clave MS (MS Product Key) (es decir eProdKeyType == SL_PRODKEY_CUSTOM), el solicitante tiene que completar su información propia de la clave del producto. Si el producto utiliza la clave de producto MS (MS Product Key), entonces la información del producto clave personal (el CustomPrdKeyInfo) puede ser no ignorada.

typedef del struct _tagSL_CUSTOM_PRODKEY_INFO

DWORD dwSKUID //identificación única para SKU específico, por ejemplo identificación de grupo en MS PID.

DWORD dwSerialNumber; //número de serie único, ej. número del canal + de serie en el MS PID.

} SL_CUSTOM_PRODKEY_INFO;

[0051] El número de versión actual de SL_PRODKEY es 1. El solicitante puede utilizar SL_CURRENT_PRODKEY_VERSION en el campo del dwVersion.

dwNumOfApps

[0052] [in - entrada] El número de la aplicación GUID en pguidApps

pguidApps

[0053] [in - entrada] Una lista de aplicación de GUID. La aplicación representa la aplicación de para que siendo instalada la licencia. Por ejemplo, el programa de inicio de Office puede llamar esta función para instalar las licencias de Word, Excell especificando cada aplicación GUID en pguidApps. pguidApp no puede ser NULO aquí.

dwNumOfApps



75

154
153

[0054] [in - entrada] Número de archivos de licencia archiva.

ppszLicFile

[0055] [in - entrada] Nombres de archivo en arreglos de cadenas (array of strings).

Returns

[0056] Éxito o falla

SLUninstall

[0057] La función SLUninstall desinstala una licencia de producto de una aplicación

HRESULT SLUninstall (

HSLC *hSLC,*

CONST SL_PRODKEY * pAppPrdKey

);

Parameters (Parámetros)

hSLC

[0058] [in - entrada] Administración del actual manejo del contexto del cliente SL

pAppPrdKey

[0059] [in - entrada] Vea arriba la definición en la conexión con SLInstall.

Observaciones

[0060] Una aplicación podría tener licencias de más de un producto. Por ejemplo, cuando la un usuario desinstala la suit de Office (Conjunto de Programas de Office), la asociación entre la licencia de la suit de Office y WinWord deben ser removidos, pero la licencia de WinWord independiente no debe ser removida.

[0061] Cuando tiene éxito SLUninstall:

[0062] La información asociada con la clave del producto es removida. (véase SLInstall, arriba, para la información asociada).

[0063] La asociación de claves de producto con la aplicación GUID es retirada.

[0064] Los archivos de la licencia asociados al producto GUID preferiblemente todavía se guardan.

Returns

[0065] Éxito o falla

SLConsumeRight

[0066] La función de derecho de consumo SLConsumeRight deja a una aplicación examinar o ejercitar los derechos en una licencia local almacenada. Llamar esta función ata una licencia al derecho mencionado en pszRightName. Si este derecho no se puede ejercitar por el solicitante actual, entonces la aplicación falla. Si la función tiene éxito, la acción



27

asociada al derecho puede ser ejecutado (como disminuyendo un contador de uso que disminuye, la cuenta de tiempo, o nada)

HRESULT SLConsumeRight(

HSLC *hSLC,*

PCWSTR *pszRightName,*

SL_ASYNC_CONTEXT * *pAsyncContext*

);

Parameters

hSLC

[0067] [in - entrada] Administración del actual manejo del contexto del cliente SL

pszRightName

[0068] [In - Entrada] El nombre de los derechos necesita ser evaluado. En el diseño actual, el nombre de derecho es definido por las aplicaciones. El SL abre la licencia y evalúa la condición basada en el nombre de derecho.

pAsyncContext

[0069] [in/out – entrada/salida] Si el *pAsyncContext* es NULO, esta función trabaja en modo síncrono, sino, la función trabaja en modo asíncronico. SL_ASYNC_CONTEXT es transparente al solicitante y es manejado por SLC.

Observaciones

[0070] Todas las licencias asociadas con la aplicación GUID (especificado en SLOpen) serán combinadas conceptualmente en una licencia lógica.

[0071] Si son garantías consumibles múltiples del derecho, entonces la licencia con una prioridad más alta será consumida primero.

Returns - Retorna

[0072] Éxito o falla

SLInitializeAsyncContext

[0073] La función SLInitializeAsyncContext inicializa el contexto asíncronico para que las funciones de SLC hagan llamada asíncronica.

HRESULT SLInitializeAsyncContext(

SL_ASYNC_CONTEXT * *pAsyncContext*
asíncronico

HANDLE *hEvent,*

PVOID *pvReserved*

);

Parameters - Parámetros

pAsyncContext



159

75

156
155

[0074] [in/out – entrada/salida] Apuntador al contexto asincrónico que contiene la información asincrónica de la llamada.

hEvent

[0075] [In - Entrada] El objeto del evento usado para la sincronización.

pvReserved

[0076] [in -entrada] Reservado para la extensión.

Returns - Retorna

[0077] Éxito o falla.

SLCancelAsyncCall

[0078] La función de SLCancelAsyncCall se utiliza para cancelar una llamada asincrónica.

HRESULT SLCancelAsyncCall(

SL_ASYNC_CONTEXT * pAsyncContext, // Contexto
asincrónico

BOOL fAbortCall // cancela
inmediatamente

);

Parameters - Parámetros

pAsyncContext

[0079] [In -entrada] Contexto asincrónico para la llamada asincrónica del SL.

fAbortCall

[0080] [In -Entrada] Si es VERDADERO, la llamada es cancelada inmediatamente. Si es FALSO, espera el SL para terminar la llamada.

Observaciones

[0081] Hay dos maneras para que el solicitante requiera la cancelación de una llamada-abortiva y nonabortiva asincrónica. En una cancelación abortiva (fAbortCall es VERDADERO), la función de SLCancelAsyncCall envía una notificación de cancelación al SLC y la llamada asincrónica es cancelada inmediatamente, sin esperar una respuesta del SLC. En una cancelación no-abortiva (fAbortCall es FALSO) la función de SLCancelAsyncCall notifica al SLC de la cancelación y el solicitante espera SLC para terminar la llamada.

Returns - Retorna

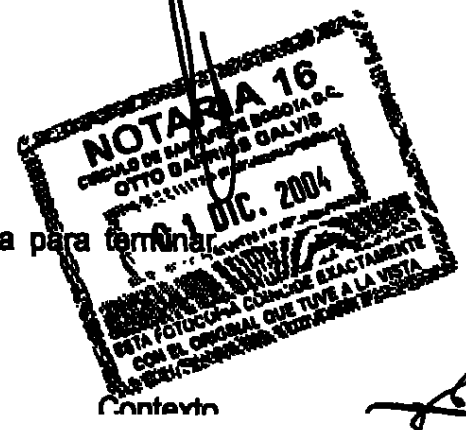
[0082] Éxito o falla

SLCompleteAsyncCall

[0083] La función SLCompleteAsyncCall se utiliza para terminar una llamada asincrónica de SLC.

HRESULT SLCompleteAsyncCall(

SI ASYNC_CONTEXT * pAsyncContext //



[Handwritten signature]

156

asincrónico

HRESULT * *phrAsyncCall* // código de error de la llamada asincrónica sometida

};

Parameters - Parámetros

pAsyncContext

[0084] [In – entrada] Contexto asincrónico para la llamada asincrónica del SL.

phrAsyncCall

[0085] [Salida] El código de error de la llamada asincrónica sometida.

Observaciones

[0086] Si el solicitante llama esta función antes de que haya llegado la respuesta, la llamada retorna **E_SLC_ASYNC_CALL_PENDING**. El almacenador intermediario debe ser válido y debe ser bastante grande recibir el valor de vuelta. Si la llamada no vuelve **E_SLC_ASYNC_CALL_PENDING**, esta invocación de **SLCompleteAsyncCall** es final para la llamada asincrónica. Después de esta llamada de función, sin importar éxito o falla, todos los recursos asignados para esta llamada asincrónica se liberan. (las llamadas subsecuentes a las funciones de **SLCompleteAsyncCall** o de **SLCancelAsyncCall** tiene resultados indefinidos hasta que una nueva llamada sobre la estructura de **SL_ASYNC_CONTEXT** sea iniciada).

Returns – Retorna

Valor	Significado
S_OK	La llamada fue terminado con éxito
E_SLC_INVALID_ASYNC_CONTEXT	El contexto asincrónico de la llamada No es inválido
E_SLC_ASYNC_CALL_PENDING	La llamada todavía no ha terminado
E_SLC_CALL_CANCELLED	La llamada fue cancelada

SLGetInformation

[0087] La función de **SLGetLicenseInfo** se utiliza para conseguir una variedad de información.

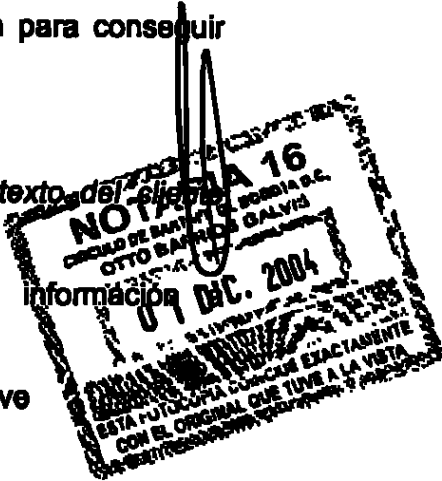
HSLC de HRESULT **SLGetInformation**(

SL HSLC *HSLC*, // Manejo del contexto del SL

DWORD *dwCategory*; // La categoría de la información recuperar

PCWSTR *pszKeyName*, // Nombre de la clave

DWORD* *pdwType*, // Tipo de valor



97

157

SIZE_T* *pcbValue*, // Tamaño del valor
PBYTE * *ppbValue*, // Indicador del almacenador intermedio
del valor
);

Parameters – Parámetros

hSLC

[0088] [In - Entrada] Administración del actual manejo del contexto
del cliente SL

dwCategory

[0089] [In - Entrada] La categoría de la información.

Categoría

Significado

SL_CAT_RIGHTDATA

Obtiene la información del derecho
atado. La licencia tiene que ser
consumida con éxito antes de conseguir
estos datos de derecho.

SL_CAT_DYNAMICPROPERTY

Obtiene la información que no está en la
licencia pero calculado en el tiempo de
ejecución. Por ejemplo, (Período de días
de *Gracia* *Restantes* -
RemainingGracePeriodDays. El derecho
tiene que ser consumido antes de la
llamada.

Name

Significado

RemainingGracePeriodDays: DWORD

El período de gracia se define en una
licencia fuera-de-caja (out-of-box). Una
vez que la aplicación es instalada, el
tiempo está contando hacia tras. Las
aplicaciones pueden comprobar períodos
de gracia restantes después de que
hayan consumido la licencia.

ActivationStatus: DWORD

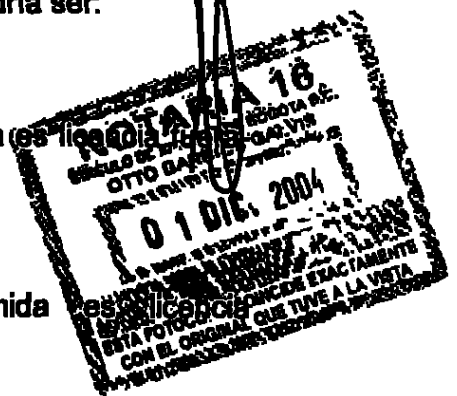
Después de que las aplicaciones
consumieron la licencia, puede
conseguir el tipo de licencia consumido.
El valor de retorno podría ser.

SL_LIC_OOB

La licencia consumida es licencia
de-caja (out-of-box).

SL_LIC_ACQUIRED

La licencia consumida
adquirida.



78

150
158

SL_LIC_NONE

No hay licencia disponible.

SL_CAT_SERVICEINFO

Obtiene la información que no depende de la licencia.

El solicitante puede conseguir esta categoría de a información sin consumir la.

Nombre

Significado

SLVersion: DWORD

La versión del formato SL.1.2.3.4.

HWID: BINARY

HWID Actual

SL_CAT_WINDOWSINFO

Obtiene la Información que es una característica del derecho atado en la licencia de Windows. Esto es para compensación. La licencia de Windows ha sido consumida por el servicio SL y el servicio SL guarda las características de derechos atados.

SL_CAT_ENUMLICINFO

Cuando el SLEnumLicenseis es llamado y tiene éxito, el solicitante puede preguntar la información de la licencia enumerada usando esta categoría.

pszKeyName

[0090] [in - Entrada] El nombre de la Clave. E.g. BuildNumber (Construye Número)

pdwType

[0091] [out - salida] Tipo de Datos

Value

SL_DATATYPE_SZ

SL_DATATYPE_DWORD

SL_DATATYPE_BINARY

Meaning

Cadena única

DWORD

Binary (Binario)

pcbValue

[0092] [out - salida] Tamaño de la memoria intermedia asignada (en bytes).

ppbValue

[0093] [out - salida] Si tiene éxito, el dato es retornado en la memoria intermedia asignada por el SLC. El solicitante no tiene que llamar a SLFreeMemory para liberar memoria.

Returns - Retorna

[0094] Éxito o Falla



Handwritten signature or mark.

160
159

SLAcquireLicense

[0095] La función de SLAcquireLicense se utiliza para adquirir la licencia en línea para el usuario. SLC enumera las claves del producto asociadas a la aplicación y toma la clave del producto con la prioridad más alta del producto (véase SLInstall, información del registro). Después el SL consigue el URL de la cámara de compensación de licencia fuera-de-caja (out-of-box) y conecta con la cámara de compensación para conseguir una licencia.

[0096] SLAcquireLicense podría ser un proceso muy largo. Las aplicaciones pueden llamar esta función en modo asíncrono especificando el pAsyncContext (NULO significa modo síncrono).

HRESULT SLAcquireLicense(
 HSLC *hSLC,* // Manejo del contexto cliente SL
 PCWSTR *pszProdKeyHash* // Clave del producto desmunaza

 PCWSTR *pszPublishLicense* // Cadena de la licencia de
publicación

SL_ASYNC_CONTEXT * pAsyncContext

);

Parameters – Parámetros

hSLC

[0097] [In - entrada] Administración del actual manejo de contexto del cliente SL

pszProdKeyHash

[0098] [in - entrada] Cadena de la clave del producto separada. La separación de la clave es creada cuando SLInstall es llamado y mantenido por el servicio de licencia.

PszPublishingLicense

[0099] [In - Entrada] Secuencia de la licencia de publicación.

pAsyncContext

[0100] [In - Entrada] Contexto asíncrono para la llamada asíncrona del SL.

Observaciones

[0101] La licencia adquirida será almacenada en almacén de la licencias por consiguiente y la información de la licencia será registrada, también (véase SLInstall)

[0102] Las aplicaciones podrían necesitar agregar más información del cliente a la licencia. La aplicación puede poner la información al pbAppData llamada y estos datos serán enviados a la cámara de compensación.

[0103] Cuando esta función tiene éxito:

[0104] Envío de la información obligatoria necesaria al servidor especificado de la licencia.



80

161
160

[0105] Recibe la licencia del servidor de licencia.

[0106] Almacenó la licencia en almacén de la licencia. Vea la descripción de SLInstall, arriba, de cómo se almacena el archivo de la licencia.

Returns - Retorna

[0107] Éxito o falla.

SLGenerateTextChallenge

[0108] Genera e instala el texto requerido para que se encamine a un emisor de la licencia fuera de la forma de banda (teléfono, email, archivo compartido, etc).

```
HRESULT SLGenerateTextChallenge(  
    HSLC      hSLC,           // Manejo del contexto cliente SL  
    PCWSTR    pszProdKeyHash // Clave del producto detallada  
  
    BOOL      fSingleSession, // Sesión sencilla  
    solamente?  
  
    PWSTR *ppszChallenge // Apuntador de memoria  
    intermedia para contener el texto exigido  
);
```

Parameters - Parámetros

hSLC

[0001] [in - entrada] Administración del actual manejo de contexto del cliente SL
pszProdKeyHash

[0002] [in - entrada] Cadena de la clave del producto detallada. La clave del producto detallada es creada cuando SLInstall es llamada y mantenida por servicio de licencia.

bSingleSession

[0003] [in - entrada] Especifica si la respuesta de texto correspondiente del elemento de la licencia será válido o no solo para el tiempo de vida del manejo de la sesión SLC.

ppszChallenge

[0004] [out - salida] Si tiene éxito, el texto exigido es retomado en la memoria intermedia asignada por el SLC. El solicitante necesita llamar a SLFreeMemory para liberar la memoria asignada.

Returns - Retorna

[0005] Éxito o Falla.

SLDepositTextResponse

[0114] Deposita la respuesta a un texto exigido de la instalación en el sistema que licencia. Utilizado para activar una licencia con un código de acceso condicional. Solamente válido si hay una exigencia excepcional del texto que se ha publicado para una licencia. Si la licencia original especificaba que el desafío era solamente válido para una sola sesión, este API se debe llamar con la respuesta antes de que el manejo de SLC sea cerrado o la entrega de la respuesta fallará.

HRESULT SLDepositTextResponse(



81

162
161

HSLC *hSLC* // Manejo del contexto del cliente SL

PCWSTR *pszProdKeyHas,*

PCWSTR *pszResponse* // Memoria Intermedia que contiene el texto de respuesta
);

[0115] [in - entrada] Administración del actual manejo de contexto del cliente SL

pszProdKeyHash

[0116] Cadena de la clave del producto detallada.

pszChallenge

[0117] [in - entrada]

Returns – Retorna

[0118] Éxito o Falla.

SLEnumLicense

[0006] El SLEnumLicensefunction se utiliza para enumerar licencias instaladas y para conseguir la información de la licencia.

HRESULT SLEnumLicense(
 HSLC *hSLC,* // Manejo del contexto del cliente SL
 CONST GUID* *pguidApp,* // Aplicación GUID
 DWORD *dwIndex* // Número índice
);

Parameters - Parámetros
hSLC

[0120] [in - entrada] Administración del actual manejo de contexto del cliente SL

pguidApp

[0121] [in - entrada] Vea SLInstall. Si el pguidApp no es NULO, entonces las licencias asociadas con este GUID se enumeran. Si el GUID es NULO, entonces todas las licencias se enumeran.

dwIndex

[0122] [in - entrada] El número de índice de la licencia a ser recuperado. Este valor debe ser cero para la primera llamada a la función de SLEnumLicense después incrementada para las llamadas subsecuentes.

[0123] La función puede retomar licencias en cualquier orden

Returns - Retorna

[0124] **E_SL_N _MORE_DATA** - Ninguna licencia en el índice especificado.



162

Observaciones

[0125] Si SLEnumLicenses tuvo éxito, la información seleccionada de la licencia puede ser alcanzada llamando SLGetInformation y la categoría es SL_CAT_ENUMLICINFO

Ejemplo:

```
DWORD i=0;
```

```
PBYTE PbProductPriority = NULL; // NULO
```

```
PBYTE PbRemainingGracePeriodDays = NULL; // NULO
```

```
For (i=0; ; i++)
```

```
{  
EXIT_ON_ERROR(SLEnumLicense(hSLC, NULL, dwIndex));
```

```
if (E_SL_NO_MORE_DATA==SCODE(hr))
```

```
hr = S_OK;
```

```
break;
```

```
}
```

```
EXIT_ON_ERROR(SLGetInformation(hSLC, SL_CAT_ENUMLIC,  
"ProductPriority", &dwType, &cbProductPriority, &pbProductPriority));
```

```
EXIT_ON_ERROR(SLGetInformation(hSLC, SL_CAT_ENUMLIC,  
"RemainingGracePeriodDays", &dwType, &cbRemainingGracePeriodDays,  
&pbRemainingGracePeriodDays));
```

```
Exit:
```

```
SLFreeMemory(pbProductPriority);
```

```
SLFreeMemory(pbRemainingGracePeriodDays);
```

```
}
```

SLFreeMemory

[0007] La función de SLFreeMemory se utiliza para liberar la memoria asignada por SLC.

```
VOID SLFreeMemory(  
PVOID pvMemBlock, // Apuntador a la memoria  
);
```

Parameters - Parámetros

pvMemBlock

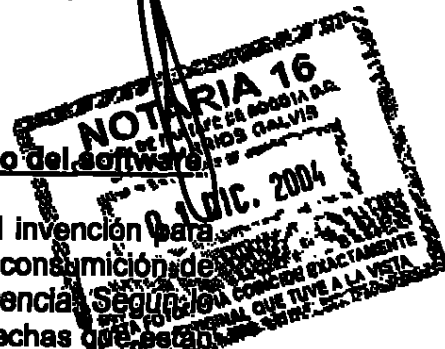
[0008] [in - entrada] Bloque de memoria previamente asignado para ser liberado.

Returns - Retorna

[0009] Ninguna

El uso del Licenciamiento de Software API para controlar el uso del software

[0129] Un producto de software utiliza el API de la actual invención para varios propósitos relacionados con licenciamiento, incluyendo la consumición de los derechos en una licencia, y la recuperación de datos de la licencia. Según lo observado arriba, el API permite que el software determine los derechos que están presentes en la licencia, pero, preferiblemente, está el software para determinar qué hacer con esa información - e.g., la concesión o negación del acceso a una



23

164
163

característica, cesa la operación en conjunto, etc. La descripción siguiente de las figs. 4 y 5 demuestran cómo el API de la actual invención es utilizado por un producto de software.

[0130] La fig. 4 muestra un proceso de ejemplo por el cual una aplicación "consume" un derecho. La aplicación llama el método SLConsumeRight (402). Según lo discutido arriba, los argumentos de la función SLConsumeRight incluyen el manejo del cliente asignado por el servicio de licenciameinto, y el nombre del derecha (que es asignado por el vendedor del software al cual pertenece el derecha). El servicio de licenciamiento (servicio 202, mostrado en la fig. 2) recibe la llamada (404). El servicio entonces localiza las licencias que contienen el derecho, y comprueba los enlaces y la validez de las licencias. Según lo observado arriba, la licencia está situada en el almacén de licencia; si hay más de una licencia que pertenece al software de aplicación al cual pertenece la llamada SLConsumeRight, entonces una regla de prioridad se puede utilizar para seleccionar una de las licencias aplicables. La comprobación de los enlaces significa que: (1) la licencia está limitada a la clave del producto de la aplicación identificada por el manejo del cliente; y (2) la licencia está limitada a la máquina en la cual el software está funcionando (o al grupo máquinas de las cuales la máquina actual es un miembro). La comprobación de validez puede incluir la determinación de que no ha expirado el derecho (en la caja de licencias que especifican una fecha de vencimiento), y de que el número máximo de derecho de las aplicaciones no está excedido (en el caso donde la licencia especifica un número máximo de veces que se puede utilizar el derecho (es decir, "consumido")).

[0131] Si se encuentran la licencia y/o el derecho de ser limitados correctamente y válidos (408), después la licencia está limitada al derecho solicitado en la llamada del API (412). (debe ser observado que "atar" una licencia a una máquina, al ambiente, y a una clave del producto significa que la licencia especifica con qué calve de la máquina(s) y del producto puede ser utilizada; "atar" una licencia a un derecho significa que la función de consumo ha sido acertada, y el derecho de una licencia particular se está consumiendo. A través de esta descripción, estará claro el contexto de que el significado de "atar" aplica.) La llamada del API después retorna a la aplicación que llama e indica que la llamada era acertada (414). Si se ha encontrado la licencia y/o el derecho para ser inválidos, o no correctamente atado a la máquina, al ambiente, o a la identificación del producto, entonces el retorno de la llamada de SLConsumeRight a la aplicación que llama e indican que la operación falló (410).

[0132] Si la llamada de SLConsumeRight retorna con una falla, después el derecho especificado en la llamada de una licencia no se puede consumir, y no hay información sobre ese derecho disponible para la aplicación que llama. Sin embargo, si el derecho se consume con éxito, después la aplicación puede utilizar el enlace del derecho de la licencia para conseguir la información de la licencia sobre el derecho. Por ejemplo, una licencia puede contener un derecho general llamado "funcionamiento," que indica que la aplicación puede ser ejecutada. Sin embargo, por derecho de "funcionamiento", la licencia puede contener parámetros más específicos sobre el uso la aplicación - e.g., la licencia puede especificar si las características particulares de una aplicación (e.g., la impresión, edición, grabado, etc.) deberían estar encendidas o apagadas, y puede dar los parámetros específicos para el uso de estas características (e.g., el documento se puede grabar solamente en las máquinas que están funcionando en un dominio particular, o la característica de la impresión se puede utilizar solamente por treinta días, etc.). El API SL no requiere ningún tipo particular de información asociado a un derecho, sino que proporciona algún mecanismo por el que el vendedor de aplicación puede asociar cualquier tipo de información a un derecho que se puede después recuperar e interpretar por la aplicación.

[0133] Si se asume que una derecha se ha consumido con éxito según lo descrito en fig. 4, la aplicación puede entonces recuperar la información asociada al derecho. El proceso de recuperar esta información se describe en la fig. 5



24

169

[0134] Primero, la aplicación llama el método SLGetInformation en el derecho enlazado (502). Varios tipos de información que pueden ser recuperados se describen arriba en la conexión con la descripción del método SLGetInformation. El servicio de licenciamiento entonces recibe la llamada (504). El servicio recupera la información solicitada del archivo de la licencia que contiene la derecha encuadrada (506). El servicio de licenciamiento entonces pone esta información en una memoria intermedia (508), y retorna a la aplicación que llama (510). La aplicación que llama entonces lee el contenido de la memoria intermedia, y realiza cualesquiera acciones si se juzga necesario basado en la información recuperada.

[0135] Debe ser observado que el servicio de licenciamiento puede no estar enterado del significado de la información que está dirigiendo como parte de una llamada de SLGetInformation. Según lo discutido arriba, el sistema de licenciamiento proporciona un mecanismo por el cual un vendedor de software puede crear los derechos, y puede asociar la información a los derechos. La invención no se limita tampoco a ningún tipo particular de información que se puede asociar al derecho. Cuando la información se recupera de la licencia, es pasada simplemente por el servicio de licenciamiento a la aplicación en una memoria intermedia (Buffer). La aplicación entonces interpreta la información recuperada, decide qué acciones se tomarán basadas en esa información, y utiliza sus propias características de seguridad para hacer cumplir la decisión de la aplicación. (e.g., si, basado en la información recuperada, la aplicación decide inhabilitar la característica de impresión, la aplicación contiene el código que inhabilita realmente esta característica y, posiblemente, el código que evita que un hacker trate de forzar inhabilitar la característica de impresión.)

[0136] Se observa que los ejemplos precedentes se han proporcionado simplemente con el fin de explicación y no es de manera alguna para ser interpretado como limitación de la actual invención. Mientras que la invención se ha descrito referente a varias personalizaciones, se entiende que las palabras que se han utilizado aquí son palabras de descripción y de ilustración, más bien que palabras de limitación. Además, aunque la invención que se ha descrito aquí referente a medios, a materiales y a personalizaciones particulares, la invención no se piensa para ser limitada a los detalles divulgados aquí; más aún, la invención se extiende a todas las estructuras funcionalmente equivalentes, métodos y aplicaciones, por ejemplo están dentro del alcance de las demandas añadidas. Los expertos en la materia, teniendo la ventaja de las enseñanzas de esta especificación, pueden efectuar modificaciones numerosas y los cambios pueden ser realizados sin salir del alcance y del espíritu de la invención en sus aspectos.

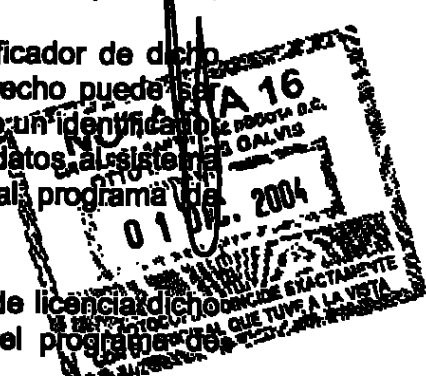
Qué es Demandado:

1. Un sistema para apoyar la obligación de una licencia para un programa de computadora, el sistema comprende:

un componente de licenciamiento que mantiene un almacén de licencia en el cual se almacena la licencia, la licencia que comprende un derecho en el software y un sistema de datos se asocian al derecho dicho, el componente de licenciamiento que exponía una interfaz accesible al programa de computadora, dicha llamada interfaz comprende:

un método de derecho de consumo que recibe un identificador de dicho derecho del programa de computadora y se determina si el derecho puede ser ejercitado; y un método de recuperación de información que recibe un identificador de dicho derecho del programa de computadora y proporciona datos al sistema dicho, o información basada en el sistema dicho de datos, al programa de computadora.

2. El sistema de la demanda 1, en donde el componente de licenciamiento es usable por una pluralidad de programas de computadora, el programa de



85

165
165

computadora que es incluido entre la pluralidad dicha de programas de computadora, en donde dicha llamada interfase abarca:

Un método de apertura de manejo que proporciona al programa de computadora; en donde el método de consumo de derechos recibe el manejo del programa de computadora y utiliza el manejo para identificar el programa de computadora del cual una llamada al método de consumo de derechos se recibe.

3. El sistema de la demanda 1, en donde la licencia es una de una pluralidad de licencias que se almacenen en dicho almacén de licencia, y en donde el método del derecho de consumo hace que el componente de licenciamiento seleccione la licencia basada en unos o más factores que comprende:

si el almacén de la licencia está asociado al programa de computadora; y una regla del conflicto que determina qué licencia seleccionar entre una pluralidad de licencias que se asocian al programa de computadora.

4. El sistema de demanda 1, en donde dicha llamada interfase comprende: un método iniciador de contexto asincrónico que establece un contexto asincrónico que procesa y proporciona un identificador de contexto de dicho programa de computadora; en donde el método dicho del consumo de derecho recibe el identificador del contexto dicho del programa de computadora dicho y procesa una petición del consumo de derecho asincrónicamente en respuesta al recibo del identificador del contexto dicho.

5. El sistema de demanda 1, en donde el método de los derechos de consumo determinan si el derecho puede ser ejercido basado en si el derecho está identificado en la licencia.

6. El sistema de demanda 1, en donde el programa de computadora y el componente de licenciamiento se ejecutan en una máquina, y en donde el método del derecho de consumo determina si el derecho puede ser ejercido basado en si la licencia está limitada a dicha máquina.

7. El sistema de demanda 1, en donde el programa de computadora se asocia a un identificador del producto, y en donde el método del derecho de consumo determina si el derecho puede ser ejercido basado en si la licencia está limitada a dicha máquina o a una clase de máquinas de las cuales dicha máquina es un miembro.

8. Un método de restringir el uso de un programa de computadora asociado con una licencia, la licencia que especifica un derecho en el programa de computadora, el método comprende:

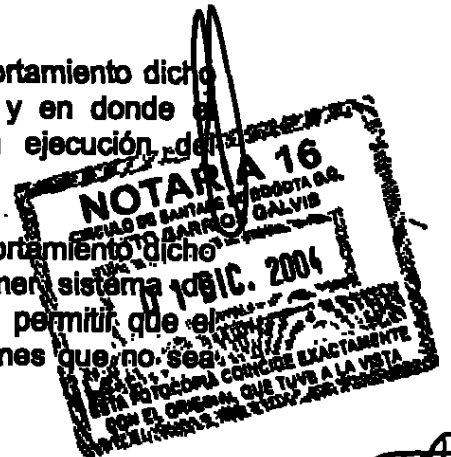
Invocando un servicio de licenciamiento haciendo una primera llamada a un primer método de interfase del servicio de licencia mencionado, dicha primera llamada siendo parametrizada por un identificador asociado con el derecho mencionado;

en respuesta a la dicha primera llamada que recibe una indicación como si el derecho es posible; y

ajustando a un primer comportamiento o a un segundo comportamiento según la indicación.

9. El método de la demanda 8, en donde el primer comportamiento dicho abarca permitir que el programa de computadora se ejecute, y en donde el segundo comportamiento dicho abarca la interrupción de la ejecución del programa de computadora.

10. El método de la demanda 8, en donde el primer comportamiento dicho abarca permitir al programa de computadora realizar un primer sistema de funciones, y en donde el segundo comportamiento dicho abarca permitir que el programa de computadora realice un segundo sistema de funciones que no sea idéntico al primer sistema dicho de funciones.



50

11. El método de la demanda 8, en donde el derecho se asocia a un sistema de datos, en donde el método además abarca:

haciendo una segunda llamada a un segundo método de la interfase dicho, el segundo método dicho siendo parametrizado por una indicación del derecho; y en respuesta a la segunda llamada dicha, recibiendo el sistema dicho sistema de datos.

12. El método de la demanda 11, además comprende:

la dirección de la operación del programa de computadora basado en el mencionado sistema de datos.

13. El método de la demanda 8, además comprende:

haciendo una segunda llamada a un segundo método de la mencionada interfase; y

en respuesta a la segunda llamada dicha, recibiendo un manejo;

en donde la segunda llamada dicha se hace antes de la primera llamada dicha, y en donde la primera llamada dicha es además parametrizada por el mencionado manejo.

14. El método de la demanda 8, además comprende:

haciendo una segunda llamada a un segundo método de la mencionada interfase; y

en respuesta a la segunda llamada dicha, recibiendo un contexto asincrónico; en donde la segunda llamada dicha se hace antes de la primera llamada dicha, en donde la primera llamada dicha es además parametrizada por el contexto asincrónico dicho, y en donde el programa de computadora realiza por lo menos una acción mientras que la primera llamada es manejada asincrónicamente.

15. El método de la demanda 8, en donde el primer método dicho determina si el derecho es ejercible basado en uno o más factores que comprenden:

si la licencia está limitada a una máquina o a un ambiente en los cuales el programa de computadora se esté ejecutando;

si la licencia o el derecho está limitado a un identificador del producto asociado al programa de computadora;

si ha expirado la licencia o el derecho; y

si el derecho se ha consumido un número de veces en exceso del derecho especificado en la licencia.

16. El medio legible por computador que se codifica sobre las instrucciones ejecutables de la computadora para realizar que un método permita la aplicación de una licencia a un programa de computadora, el método comprende:

recepción de una primera llamada del método del programa de computadora, la primera llamada del método que identifica un derecho en el programa de computadora;

determinando que el derecho está contenida en la licencia y es ejercible; y volviendo al programa de computadora una indicación de que el derecho es ejercible.

17. El medio legible por computador de la demanda 16, en donde la indicación abarca un enlace del derecho a la licencia.

18. El medio legible por computador de la demanda 16, en donde el acto de determinación dicho se basa encendido si la derecha está especificada en la licencia.

19. El medio legible por computador de la demanda 16, en donde el acto de determinación dicho basado en si la licencia está limitada a una máquina en la cual el programa de computadora se esté ejecutando



167

20. El medio legible por computador de la demanda 16, en donde el acto de determinación dicho basado en si la licencia o el derecho está limitado al programa de computadora.

21. El medio legible por computador de la demanda 16, en donde el acto de determinación dicho basado en si no-está expirada la licencia o el derecho.

22. El medio legible por computador de la demanda 16, en donde el acto de determinación dicho basado en si la licencia se ha consumido un número de veces que excede un límite.

23. El medio legible por computador de la demanda 16, en donde el método además comprende:

recibiendo una segunda llamada del método del programa de computadora; y en respuesta a la segunda llamada del método, retomando un manejo al programa de computadora que identifica el programa de computadora;

en donde la primera llamada dicha del método es subsecuentemente realizada a la segunda llamada dicha del método, y en donde la primera llamada dicha del método además identifica dicho manejo.

24. El medio legible por computador de la demanda 16, en donde el método además comprende:

recibiendo una segunda llamada del método del programa de computadora; en respuesta a la segunda llamada del método, retomando un contexto asincrónico al programa de computadora, en donde la primera llamada del método es subsecuentemente ejecutada a la segunda llamada del método e identifica el contexto asincrónico dicho; y

ejecutando la primera llamada del método asincrónicamente mientras que el programa de computadora realiza una acción.

25. El medio legible por computador de la demanda 16, en donde el derecho se asocia a un sistema de datos, y en donde el método además comprende:

recibiendo una segunda llamada del método que indica el derecho; y

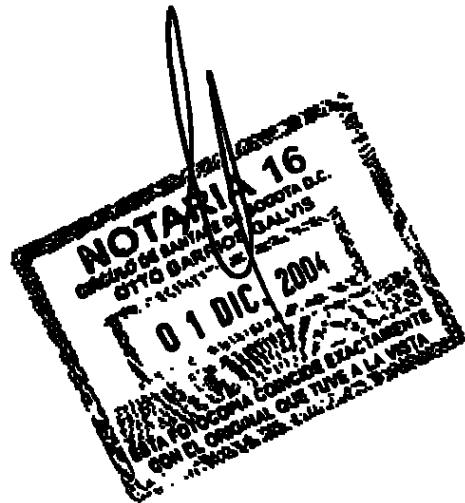
en respuesta a la segunda llamada dicha del método, proporcionando el sistema de datos al programa de computadora.



8E

RESUMEN DEL DESCUBRIMIENTO

Un licenciameinto de software API (Application Program Interface – Interfase del Programa de Aplicación) que permite que los productos de software utilicen la funcionalidad de la administración de licencia de un servicio común. Una licencia especifica los derechos en un producto de software. El producto de software llama un método de consumo en el API para consumir un derecho. Si existe el derecho, el servicio enlaza el derecho a la licencia en la cual se encuentra el derecho. El producto de software hace cumplir los términos de la licencia concediendo, o negando, el acceso a alguna o todas las características dependiendo de si uan instancia válida del derecho es encontrada. Los datos arbitrarios se pueden asociar a un derecho. El API incluye un método para recuperar datos de un derecho que ha sido limitado previamente por el método de consumo.



120

169

SOLICITUD DE PATENTE

DECLARACION Y PODER DE REPRESENTACION LEGAL	
FICHA DEL APODERADO NO. MSFT-2822	FICHA MS NO.305442-1

Como el inventor descrito a continuación, por la presente declaro que:

La dirección de mi residencia y / o dirección postal y mi ciudadanía se especifican a continuación, al lado de mi nombre;

Yo creo que soy el primer inventor original, primero y único (cuando se especifica únicamente un nombre) o uno de los inventores originales, primeros y conjuntos (cuando si a continuación se listan nombres en forma plural) del asunto que se reclama y para el cual se busca una patente sobre el invento titulado: INTERFACE DE PROGRAMACION PARA EL LICENCIAMIENTO

Cuya especificación se adjunta con la presente, a menos que se marque la siguiente casilla:

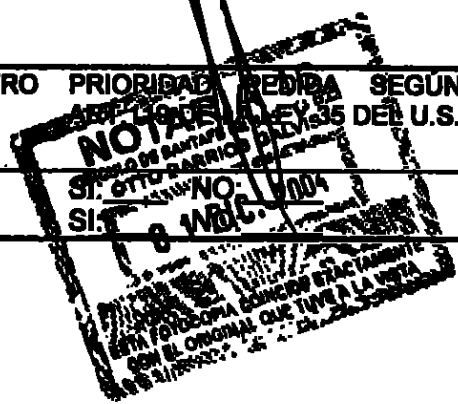
() fué presentada el _____ como No. de Solicitud Serial de los Estados Unidos o como Solicitud Internacional PCT Número _____ y fue modificada el _____ (si se aplica).

Por la presente declaro que he revisado y comprendido los contenidos de la especificación anterior, incluyendo demandas, así como modificación por cualquier modificación referida anteriormente. Reconozco el deber de exponer toda la información que sea material de patentabilidad, tal como está definido en el numeral 1.56 del artículo 37 del CFR.

Solicitud Extranjera y / o Petición de Prioridad Extranjera

Por medio de la presente hago la petición de los beneficios de prioridad extranjera de conformidad con el Artículo 119 de la ley 35 del Código de los Estados Unidos de cualquier solicitud extranjera para certificados de patente o de inventor(es), que tengan una fecha de presentación anterior a la solicitud en la cual se reclama la prioridad:

PAIS	NÚMERO DE SOLICITUD	DE	FECHA DE REGISTRO	PRIORIDAD	PEDIDA SEGUN EL
					ART. 119 DE LA LEY 35 DEL U.S.C.



Victoria Eugenia Cruz de Ramirez
TRADUCTOR E INTERPRETE OFICIAL
REG. 381 • 220402
MINISTERIO DE JUSTICIA

90

170

PODER DE REPRESENTACIÓN LEGAL:

Como el inventor mencionado, por la presente designo a los siguientes abogados y/o agentes asociados con

23377

y

David Bartley Eppenauer Reg. No.35.499	Martin L. Shively Reg. No.33.553	Ronald O. Zink Reg. No.35.744	Patricia E. Barnes Reg. No.37.038
Stacy Quan Reg. No.33.780	Jeffrey L. Ranck Reg. No.38.590	Michael W. Boclanowski Reg. No.28.692	John Weresch Reg. No.32.322

para llevar a cabo esta solicitud y para transigir en todos los asuntos dentro de la Oficina de Marcas y Patentes o en conexión con ésta.

Enviar la correspondencia a:	Dirijir las llamadas telefónicas a:
Nombre del Contacto: Steven J. Rocci Nombre de la Firma: Woodcock Washburn, LLP Dirección de la Firma: One Liberty Place, 46th Floor Ciudad, Estado y Zip: Philadelphia, PA 19103	Nombre del Contacto: Peter M. Ulman Número telefónico del Contacto: (215) 564 8276

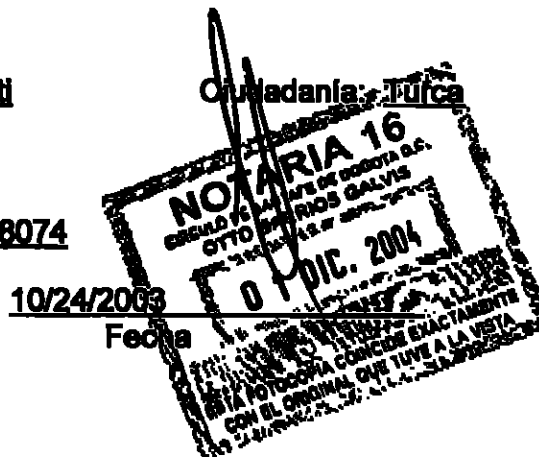
Por medio de la presente declaro que todas las declaraciones hechas dentro de ésta son ciertas hasta donde llega mi conocimiento, y que todas las declaraciones realizadas sobre la información y creencia, creo que son ciertas; y además que estas declaraciones se realizan con el conocimiento de que las declaraciones intencionalmente falsas o en forma parecida son sancionables con multas o prisión, o ambas cosas, de conformidad con la Sección 1001 del Capítulo 18 del Código de los Estados Unidos y que tales declaraciones intencionalmente falsas pueden poner en peligro la validez de la solicitud o de cualquier patente otorgada a partir de ésta.

Nombre completo del inventor: Caglar Gunakti

Residencia: 440 226th Lane NE

Dirección Postal: Sammamish, Washington, 98074

(Firma)
Firma del inventor



Victoria Eugenio Cruz de Ramirez
TRADUCTOR E INTERPRETE OFICIAL
REG. 381 • 2304/02
MINISTERIO DE JUSTICIA

Nombre completo del inventor: Wen-Pin Scott-Hsu

Ciudadanía: Taiwan

Residencia: 18114 NE 25th Street

Dirección Postal: Redmond, Washington, 98052

(Firma)
Firma del Inventor

10/24/2003
Fecha

Nombre completo del inventor: Ning Zhang
Norteamericana

Ciudadanía:

Residencia: 23808 SE 5th Street

Dirección Postal: Sammamish, Washington, 98074

(Firma)
Firma del Inventor

10/24/2003
Fecha

CERTIFICACIÓN

Yo, VICTORIA EUGENIA CRUZ DE RAMÍREZ, declaro que soy Traductora e Intérprete Oficial los idiomas Inglés y Español, con sede en la Calle 148 # 14-09 (202) en Bogotá DC, Colombia.

Declaro que la traducción adjunta es una versión fiel y completa al español de los documentos adjuntos, en idioma inglés, referentes a la solicitud y tramitación de una solicitud de Patente, con fecha de presentación 24 de octubre de 2003, ante la Oficina de Marcas y Patentes de los Estados Unidos de América.


VICTORIA EUGENIA CRUZ DE RAMÍREZ
Traductor e Intérprete Oficial
Resol. 0391-22/04/02 de MinJusticia



2° EXAMEN DE FORMA, PATENTE DE INVENCION

Radicación N° **04-94972**

Folio **173**

Concepto Técnico N° **741**

Clasificación Internacional de Patentes: **G06F 17/60**

Practicado el examen de acuerdo a lo establecido en el Art. 38 de la Decisión 486, se determinó que esta solicitud de **Patente**:

- SI ☒ NO ☐ Contiene un resumen de la invención (Art. 26-e)
- SI ☒ NO ☐ Contiene el título o nombre de la invención (Art. 27-d)
- SI ☒ NO ☐ Contiene la descripción de la invención (Art. 26-b)
- SI ☒ NO ☐ NO ES APLICABLE ☐ Contiene los dibujos necesarios para comprender la invención (Art. 26-d)
- SI ☒ NO ☐ Contiene una o más reivindicaciones (Art. 26-c)
- SI ☐ NO ☐ NO ES APLICABLE ☒ Han sido desarrollados los productos a partir de recursos genéticos o de sus productos derivados de los que cualquiera de los Países Miembros es país de origen (Art. 26-h)
- SI ☐ NO ☐ NO ES APLICABLE ☒ Los productos y o procedimientos han sido obtenidos o desarrollados a partir a partir de los conocimientos tradicionales de las comunidades indígenas, afro americanas, o locales de los Países Miembros, de acuerdo a lo establecido en la Decisión 391 y sus modificaciones y reglamentaciones vigentes (Art. 26-i)
- SI ☐ NO ☒ La invención se refiere a un producto relativo a un material biológico (Art. 26-j)
- SI ☒ NO ☐ NO ES APLICABLE ☐ La prioridad coincide con la materia reivindicada.
- SI ☒ NO ☐ **CUMPLE LOS REQUISITOS TÉCNICOS**

OBSERVACIONES Y OTROS: SI ☐ NO ☐ Ver hoja(s) anexa(s)

EXAMINADOR TÉCNICO: 202051

Bogotá D C, septiembre 30 de 2005.

2020
Bogotá D C

Doctor(a):
DARIO CARDENAS NAVAS.
Apoderado (a)

REFERENCIA:	Radicación N°	04-94972
	Trámite	002
	Evento	001
	Actuación	416
	Oficio N°	995
	Folios	001

Teniendo en cuenta que la solicitud de privilegio de patente que se tramita bajo el expediente indicado en la referencia, reúne los requisitos de forma establecidos en los artículos 26 y 27 de la Decisión 486, de la Comisión de la Comunidad Andina, y en las demás disposiciones vigentes sobre la materia, se envía el extracto de la solicitud a la oficina de comunicaciones para efecto de su publicación en la Gaceta de Propiedad Industrial, conforme lo establece el artículo 40 de la Decisión 486, a partir del 24 de abril de 2006.

Cúmplase
Dado en Bogotá D C, 04 OCT. 2005



ALIX CARMENZA CÉSPEDES DE VERGEL
Jefe División de Nuevas Creaciones.

202051