



Industria y Comercio

SUPERINTENDENCIA

Delegatura de propiedad Industrial

División de Nuevas Creaciones

SOLICITUD

PATENTE DE INVENCION

21. EXPEDIENTE No. 05-51829

54. TÍTULO MARCO DE TRABAJO PARA LA GRABACION Y
REPRODUCCION UI INTELIGENTE

51. CLASIFICACIÓN INTERNACIONAL 6 067 17/60

71. SOLICITANTE MICROSOFT CORPORATION

DOMICILIO REDMOND, WASHINGTON, ESTADOS UNIDOS DE AMERICA

74. APODERADO DARIO CALDENAS NAVAS

22. BOGOTÁ, D. C., _____

PETITORIO

Industria y Comercio
SUPERINTENDENCIA

RECEIVED - PETITION
FOLIO: 001
FECHA DE RECEPCIÓN: 05-38,822
NÚMERO DE REGISTRO: 10/882,861
DEPENDENCIA: DIVISIÓN DE PATENTES

FORMULARIO UNICO DE SOLICITUD DE PATENTE 2000-3

1

SOLICITUD DE:

☒

Patente de Invención

☐

Patente de Modelo de Utilidad

SOLICITANTE
(71)

Nombre: **MICROSOFT CORPORATION**

Dirección: **One Microsoft Way Redmond, Washington 98052, Estados Unidos de America**

Teléfono: **3123800** Fax: **3122410**

E-Mail: **general@cardenasvcardenas.com**

Lugar de Constitución: **Estado de Washington, Estados Unidos de América**

Domicilio: **REDMOND, WASHINGTON, ESTADOS UNIDOS DE AMERICA.**

IDENTIFICACION

C.C.

☐

NIT

☐

C.E.

☐

Otro

☒

Número de

REPRESENTANTE
O APODERADO
(74)

Nombre: **DARIO CARDENAS NAVAS**

Dirección: **Carrera 7 No 71-52 Torre B Piso 9**

Teléfono: **3123800** Fax: **3122410**

E-Mail: **general@cardenasvcardenas.com**

IDENTIFICACION

C.C.

☒

NIT

☐

C.E.

☐

Otro

☐

Número

17.088.829 BTA

TP **1.250 C.S.J.**

INVENTOR(ES)
(72)

Nombre: **Howard B. Herdeg, III**

Dirección: **One Microsoft Way Redmond, Washington 98052, Estados Unidos de América**

Teléfono: **3123800** Fax: **3123800**

E-Mail: **general@cardenasvcardenas.com**

Domicilio: **REDMOND, WASHINGTON, ESTADOS UNIDOS DE AMERICA.**

IDENTIFICACION

C.C.

☐

NIT

☐

C.E.

☐

Otro

☒

Título(54):

MARCO DE TRABAJO PARA LA GRABACIÓN Y REPRODUCCIÓN UI INTELIGENTE

6

Prioridad

SI

☒

NO

☐

(33) País de Origen

USA

(32) Fecha

Junio 30, 2004

(31) N° de Solicitud

10/882,861

7

Para publicar a partir de la fecha de la presente solicitud a los:

☐

6 meses

☐

12 meses

☐

18 meses

☐

Otro

Comprobante de Pago No.:

05-38,822 / 05-38,823

Mayo 26, 2005

ANEXOS

- ☒ Comprobante de pago de la tasa de presentación de la solicitud. (No. 05-38, 122)
- ☐ Comprobante de pago de la tasa por concepto de excedente de palabras en la publicación.
- ☒ Comprobante de pago de la tasa por concepto de reivindicación de prioridad. (No. 05-38, 123)
- ☒ Poderes, si fuere el caso. (Protocolo No 16.069)
- ☒ Certificado de existencia y representación legal cuando el solicitante sea persona jurídica. Protocolo No 16.069)
- ☒ Copia certificada de la primera solicitud, si se reivindica prioridad.
- ☒ Traducción simple de la primera solicitud, si se reivindica prioridad.
- ☒ Documento de cesión del inventor al solicitante o a su causante.
- ☒ Descripción de la invención.
- ☒ Una o más reivindicaciones.
- ☒ Dibujos y/o planos necesarios.
- ☐ De ser el caso, copia del contrato de acceso.
- ☐ De ser el caso, documento que acredite la licencia o autorización de uso de conocimientos tradicionales de la comunidad de las comunidades indígenas.
- ☐ De ser el caso, certificado de depósito del material biológico.
- ☒ Arte final 12x12 y 6x6 cm.
- ☒ Resumen y extracto para publicación.

10

NOMBRE: DARIO CARDENAS NAVAS



FIRMA: C.C. 17.066.629 BTA OT TP. 1.250
C.SJ.

SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO
NACIONALIDAD : COLOMBIANA / NACIONALIDAD : COLOMBIANA
FECHA : 26/05/2005 / FECHA : 26/05/2005
TITULO : MARCO DE TRABAJO PARA LA GRABACION Y REPRODUCCION DE SOFTWARE
DEPENDENCIA : DIVISION DE NUESTROS SERVICIOS

SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO

NIT : 800176089-2

RECIBO OFICIAL DE CAJA : 05 - 38,822
FECHA : MAYO 26 DE 2005

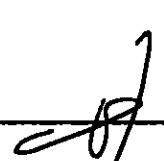
CONSIGNACION

DEPOSITANTE	TIPO PAGO	BANCO	CUENTA	No. PAGO	FECHA PGO	Vr. PAGO
CARDENAS Y CARDENAS P.I.	CONSIGNACION	BANCO POPULAR	050-00110-6	36227317	26/05/2005	1,699,000.00

C O N C E P T O

CANT.	RENTISTICO	CONCEPTO	TOTAL CONCEPTO
1	50005-01-01	SOLICITUDES	
1		TRAMITES DE SOL. DE PATENTE DE IN	443,000.00
		TOTAL :	\$ 443,000.

SON: CUATROCIENTOS CUARENTA Y TRES MIL PESOS

RESPONSABLE :  **CARDENAS & CARDENAS**

RECIBO DE CAJA APLICADO AL EXPEDIENTE No. MARCO DE TRABAJO PARA LA GRABACION Y REPRODUCCION DE SOFTWARE
GRABACION Y REPRODUCCION DE SOFTWARE

4

RECEPCION DE DINERO
RECEPCION DE DINERO
RECEPCION DE DINERO
RECEPCION DE DINERO
RECEPCION DE DINERO
RECEPCION DE DINERO
RECEPCION DE DINERO
RECEPCION DE DINERO
RECEPCION DE DINERO
RECEPCION DE DINERO

SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO

NIT : 800176089-2

RECIBO OFICIAL DE CAJA : 05 - 38.823
FECHA : MAYO 26 DE 2005

***** CONSIGNACION *****

DEPOSITANTE	TIPO PAGO	BANCO	CUENTA	No. PAGO	FECHA PAGO	Vr. PAGO
CARDENAS Y CARDENAS P.I.	CONSIGNACION	BANCO POPULAR	950-00110-6	36227317	26/05/2005	1,699,000.00

***** CONCEPTO *****

CANT.	RENTISTICO	CONCEPTO	TOTAL CONCEPTO
1	50005-01-01	SOLICITUDES	
		82 INVOCACION DE UNA PRIORIDAD EN NU	124,000.00
		TOTAL :	\$ 124,000.00

SDN: CIENTO VEINTICUATRO MIL PESOS

RESPONSABLE : CARDENAS Y CARDENAS

RECIBO DE CAJA APLICADO AL EXPEDIENTE No. MARCO DE TRABAJO PARA LA GRABACION Y REPRODUCCION DE INTELECTUAL

7

8

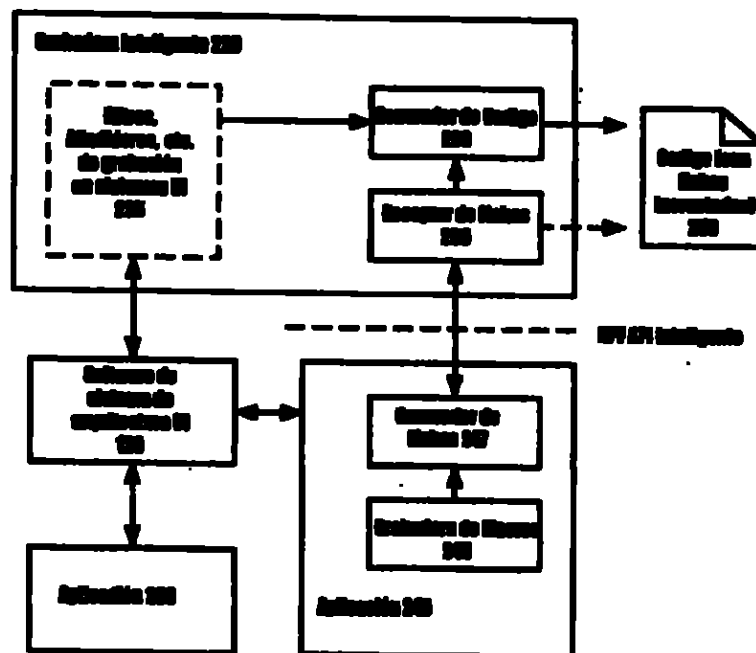
5

Art para
publicar



5

Figura 2



2

Marco de Trabajo para la Grabación y Reproducción UI Inteligente

Campo Técnico

Para grabar actividades en varias aplicaciones de Interfases gráficas de usuario se describen varias técnicas y herramientas. Por ejemplo, una herramienta de grabación que abarca todo el sistema sustenta aplicaciones de macros nativas o grabación UI como también grabaciones UI de macros no nativas.

Antecedentes

1. Interfase de Usuario Gráfica

En muchos sistemas de computación modernos, los usuarios interactúan con programas de software a través de una interfase de usuario gráfica ("GUI"). Básicamente, un GUI es una interfase entre un usuario y un computador que utiliza figuras en vez de palabras para solicitar entradas del usuario y presentar programas de salida. Un GUI típico está compuesto de elementos de interfase ("elementos UI"), que son esos aspectos de un sistema de computación o programas que se ven, se oyen, se perciben o interactúan con un usuario. Por ejemplo los elementos UI incluyen ítems tales como, iconos, botones, casillas de diálogo, casillas editables, casillas que se despliegan, casillas combo, barras de desplazamiento, listas que se pueden seleccionar, botones de arranque, botones de radio y varios componentes de Páginas Web Mundiales (por ejemplo, hipervínculos e imágenes). En un programa típico de computador es común encontrar literalmente miles de elementos UI.

Aunque un elemento individual de un GUI puede aparecerle al usuario como un ítem único, puede realmente consistir de un número de ítems separados o subelementos que serán combinados. Por ejemplo, un ítem de la barra herramientas puede consistir de un elemento de lista, un elemento de casilla, un elemento de barra de desplazamiento, etcétera. Aún más, cada uno de estos subelementos pueden estar compuestos de otros subelementos. De esta forma, los elementos UI pueden servir como bloques de construcción para construir otros elementos UI más complejos. Este enfoque es útil porque el software que administra la interfase de usuario puede reutilizar las definiciones de ciertos elementos comunes cuando los ensambla en un elemento compuesto.

Muchos elementos UI en un ambiente GUI representan rasgos de un programa y se muestran en el monitor del computador para que el usuario interactúe con el programa seleccionando, marcando, accediendo y operando con ellos. Esta interacción de usuario se hace manipulando un puntero en la pantalla (controlado típicamente por un ratón o un teclado) y presionando o haciendo clic con el botón mientras el puntero señala un elemento UI. Por ejemplo, en un procesador de palabras, un usuario puede manipular el ratón para seleccionar un ítem en la barra de menús del programa, haciendo clic en un icono de la barra herramientas, marcando un bloque del texto en la ventana de visualización. En forma similar, un usuario puede utilizar el teclado para interactuar con una aplicación de computador, por ejemplo, en un programa de procesador de palabras, un usuario puede presionar "ALT-F," o "CTRL-B," o cualquier otra combinación de teclas definidas para acceder a distintas funcionalidades del programa. Basándose en las entradas del ratón o el teclado, el computador añade, cambia, y manipula lo que se muestra en la pantalla. Tecnologías GUI suministran ambientes de usuario amigables para que los usuarios interactúen con el sistema de computación.

2. Automatización UI

La automatización UI ("UIA") es un marco de trabajo asequible para Microsoft Windows que pretende dirigirse a las necesidades de los productos de tecnología de asistencia y marcos de trabajo de pruebas automatizadas suministrando acceso programático a la Información acerca de la interfase del usuario ("UI"). Por ejemplo AUI le permite a un programa lector de pantalla acceder a la información acerca del UI de un procesador de palabras, suministrando a un programa de lectura la información que necesita para suministrar pistas audibles a un usuario con problemas de visión. A través de una serie de métodos de aplicación de programación de Interfases ("API"), del UIA suministra un mecanismo bien estructurado para crear e interactuar con el UI. Los diseñadores de controles de aplicaciones utilizan el conjunto API UIA para hacer que sus productos sean más accesibles a distintos usuarios a través de software nuevo o existente (escritos potencialmente por otras personas) para acceder a los menús del programa y a otros elementos UI. Por ejemplo, pantallas braille, lectores de pantalla (narradores), aumentadores de tamaño, y otros tipos de software pueden utilizar UIA en Microsoft Windows para facilitar al usuario el uso del computador al cual de lo contrario no tendría acceso.

En la práctica, la Automatización UI utiliza una jerarquía de elementos UI localizados en una estructura de árbol para suministrar Información UI confiable al sistema operativo de aplicaciones del computador. Los elementos de una GUI se pueden considerar como anidados dentro de ellos mismos para describir su organización en forma precisa. Por ejemplo, en la cima de una estructura de árbol hay un elemento de escritorio que es representativo del "escritorio" del GUI o área de fondo por defecto.

El elemento de escritorio tiene varios elementos de aplicación para programas de aplicación que se han invocado si están listos a ser ejecutados de acuerdo con las instrucciones del usuario (por ejemplo, un escritorio típico de Microsoft Windows

10.

poder tener varias instancias de aplicaciones tales como Microsoft Word, Microsoft Excel, etc. cargados y listos a ser ejecutados). En un nivel más bajo de la jerarquía de estructura del árbol hay marcos asociados con aplicaciones (por ejemplo, un procesador de palabras puede tener varios marcos visibles a un usuario en un determinado momento). Dentro de cada marco puede haber varios documentos, cada documento puede contener varios elementos UI (por ejemplo, botones, listas de casillas, etc). Los elementos UI en sí mismos pueden estar compuestos de otros elementos UI. Por ejemplo, una casilla de diálogo o una casilla combo puede contener otros elementos UI tales como botones de control. Aún más, los elementos de botones pueden contener más elementos UI. Esta estructura anidada puede ser arbitrariamente profunda e incluir un número arbitrario de ramas dependiendo de la interfase del usuario y sus elementos componentes.

Para algunas plataformas de sistemas operativos, una instancia de un elemento UI se asigna a un identificador para ayudar a distinguir entre un elemento UI particular de otros elementos UI. Por ejemplo, en un sistema operativo basado en Microsoft Windows, las aplicaciones se asocian con identificadores de módulo que identifican a las aplicaciones dentro de un contexto de escritorio, también algunas plataformas de interfase del usuario (por ejemplo Microsoft Windows, Swing o Java) utilizan un identificador numérico (un ID de control) para ciertos elementos UI. En algunos ambientes de computación, tales como el ambiente de Microsoft Windows, los elementos UI frecuentemente se asocian con un nombre de clase asociado a una clase de control a la que pertenece. Por ejemplo, en un sistema basado en Microsoft Windows, elementos comunes UI tales como casillas combo, listas de casillas, y botones se asocian con un nombre de clase tal como la clase ComboBox, la clase ListBox, y la clase Botón, respectivamente. En forma similar otros marcos UI pueden tener nombres para sus clases respectivas de elementos UI.

Estas técnicas identifican clases de objetos de elementos UI o tipos, pero no suministran un identificador fuerte que identifique en forma única un elemento UI a través de un reinicio del computador que corre el programa, a través de una construcción diferente del programa cuando todavía está en desarrollo, a través de abrir otra instancia del mismo programa, o para abrir el mismo programa en otro computador.

La UIA se superpone a estas deficiencias generando un ID compuesto que identifica en forma única a los elementos UI en el árbol GUI. La UIA genera un identificador compuesto añadiendo Información de Identificación (por ejemplo, un nombre de control o un tipo de control) que está asociado directamente con un elemento UI para una Información de Identificación jerárquica (por ejemplo control paterno, control de los hijos, y/o control de los hermanos) e Información de control con un patrón específico (por ejemplo, la profundidad de un elemento UI en el árbol). Por ejemplo, un identificador para un elemento UI objeto se puede generar recolectando información de identificación que se relaciona con el elemento paterno GUI que describe el arreglo jerárquico entre un elemento de páginas UI y el elemento raíz del escritorio. A través del concepto de una ruta, el identificador relacionado de la jerarquía única del elemento UI y su paternidad se pueden nivelar para identificarlo en forma única y persistente.

El identificador único (ID persistente) suministra un fácil acceso a los elementos individuales UI de tal manera que la funcionalidad del programa que alberga los elementos UI se puede programar y probar, con lo cual el elemento particular UI se puede identificar con otros módulos del programa. Para información adicional acerca de UIA ver, por ejemplo, la documentación disponible a través de la Red de Diseñadores de Microsoft.

III. Grabadoras de GUI y Constructores de Macros

La habilidad de grabar y reproducir las interacciones de un usuario con un computador en un ambiente GUI tiene la potencialidad de beneficiar a múltiples

usuarios, incluyendo a negocios (u otras organizaciones grandes), usuarios, diseñadores de software, bancos de prueba, y personal de apoyo a los computadores. Por ejemplo, las organizaciones de negocios pueden perfilar sus procesos de negocio, tales como uso de software para la administración de cadenas de distribución, automatizando gran parte del proceso. Los usuarios se benefician creando macros o libretos que combinan una serie de acciones imputadas en una acción de reproducción que se dispara con un solo paso. Los diseñadores de software pueden utilizar la habilidad para grabar las acciones de un usuario para ayudar a generar casos de prueba para software que está en desarrollo. Los probadores pueden grabar y reproducir herramientas para construir pruebas para ejecutar regresiones automatizadas de prueba. Personal de apoyo a los computadores pueden grabar las acciones de un usuario para descubrir la razón por la cual un computador se bloquea, o para ayudar a los usuarios a entender cómo utilizar el software.

A. Diseñadores Comerciales de Macros

Muchos programas convencionales de diseño de macros generan guiones que muestran los comandos internos y acciones ejecutadas por un computador para ejecutar una función en una aplicación. Sin embargo, en muchos casos, un usuario puede desarrollar guiones en forma independiente basados en una serie de comandos en los guiones y construcciones complejas de programación. Por lo tanto, los usuarios tienen que entender la lógica de programación y, hasta cierto punto, la lógica de los programas que se controlan para crear y utilizar el macro. Por ejemplo, AutoMate, un programa de macros, utiliza un diseñador de tareas que utiliza el arrastre-y-suelta para crear guiones de macro arrastrando y soltando pasos específicos en el orden en que se deben ejecutar, lo cual significa que el usuario de AutoMate tiene que entender cómo y en qué orden los comandos se deben emitir. El Macro Scheduler es un programa de creación de macros que le

permite al usuario escribir un macro utilizando más de 200 guiones de comandos y programando construcciones (que no incluyen variables actuales declaradas y otras estructuras definidas por usuario). La complejidad que se requiere para crear y editar un libreto generado por estos programas y el hecho de que los guiones generados por estos constructores de macros no representa una grabación legible de la actividad UI que le quita a esos programas de macros su utilidad, particularmente a usuario nuevos o al personal de apoyo y diseñadores de software que tratan de resolver problemas.

B. Lenguaje Interno Macro

Como una alternativa al software comercial de diseño de macros, muchas aplicaciones tienen la habilidad inherente de grabar y reproducir macros utilizando su propia aplicación especial de lenguaje de control ("lenguaje macro"). Una aplicación de lenguaje macro es típicamente única a la aplicación y generalmente está basada en el modelo de objetos interno ("IOM"). Mientras es posible construir un lenguaje de macro sin técnicas orientadas hacia los objetos, la mayoría de modelos internos utilizan una representación orientada hacia los objetos de la estructura del programa. El IOM provee un perfil o modelo de clases, atributos, operaciones, parámetros, relaciones, y asociaciones de objetos subyacentes al programa. El lenguaje macro accede a sus aplicaciones IOM y se engancha a su mecanismo de comunicación (tales como llamadas de eventos) para acceder y llamar a las distintas funcionalidades de la aplicación. Por ejemplo, un usuario de Microsoft Word puede grabar un macro para formatear texto automáticamente. Uno de los inconvenientes principales del lenguaje macro, sin embargo, es que es específico a la aplicación. Un macro que se graba en una aplicación generalmente no corre en otras aplicaciones, particularmente cuando dos de ellas han sido desarrolladas por compañías que compiten en software. En algunos casos, un

macro que se graba para una versión de una aplicación no encuentra soporte en una versión futura de la misma aplicación.

C. Dependencia en los Macros

Desarrollar macros puede ser difícil, y muchos usuarios y negocios son renuentes a hacer cualquier cosa que pueda dañar los macros existentes. Los negocios en particular son renuentes a hacer algo que pueda afectar en forma negativa sus procesos de negocios. De hecho, muchos usuarios y negocios son renuentes a actualizar el software o incluso a instalar parches por temor a a "dañar" algo. Por ejemplo, supongamos que un negocio emplea un proceso automático de negocios (por ejemplo, en la forma de un macro) que escanea los faxes que recibe para producir archivos de texto de los faxes, compara los datos en el archivo de texto a entradas en una hoja de cálculo para verificar la información de cuentas, y finalmente envía un nivel de confirmación al remitente del faxes. Este proceso de negocios utiliza generalmente un conjunto de macros potencialmente complicados y una variedad de paquetes de software para funcionar en forma apropiada (por ejemplo, un programa de escáner, un OCR, un programa de hoja de cálculo, y un programa de archivos de texto, etc.). Los negocios y los usuarios se sienten incómodos actualizando macros o software al menos de que tengan seguridad de que la inversión actual en automatización de los procesos permanecerá intacta.

D. Otras Grabadoras UI

Muchas grabadoras convencionales UI tienen inconvenientes similares a aquellos de las herramientas de macros descritos arriba en el sentido de que utilizan comandos de guiones complejos y construcciones de programación para representar los datos. Otro inconveniente es que la reproducción convencional depende de las condiciones preexistentes del computador. Por ejemplo, la

reproducción puede depender de cierta configuración de hardware, instalaciones de software, y/o el estado dinámico del ambiente de ejecución (tal como la disponibilidad o localización de elementos UI para una grabación particular de una interacción). Utilizando una herramienta de reproducción convencional, cualquier cambio a esas condiciones preexistentes pueden causar que la reproducción se detenga.

Por ejemplo, supongamos que un usuario reconfigura la interfase de una aplicación. Algunas aplicaciones basadas en GUI le permiten al usuario mover botones, reconfigurar menús, añadir o remover otros elementos UI para distintos rasgos del programa, añadir vínculos a los macros en la barra de menú, o ejecutar otras alteraciones UI. Aunque la habilidad de personalizar menús es un rasgo útil (por ejemplo permitiendo a los usuarios configurar su UI a sus necesidades específicas), puede causar muchas fallas en la grabación y la reproducción porque los elementos UI ya no están en los sitios esperados.

Por ejemplo, el grabador de macros de Microsoft Windows 3.1 intenta replicar las acciones de un usuario en un GUI grabando los movimientos del ratón y las coordenadas del ratón cuando se hace clic en uno de sus botones. En la reproducción, si un elemento GUI no está donde se espera, la reproducción falla. Para mayor información en la grabación de macros de Microsoft Windows 3.1 ver, por ejemplo, las referencias denominadas, Guía del Usuario para Trabajos en Grupo de Microsoft Windows, pág. 137. En otro ejemplo, suponga que un usuario graba una actividad GUI en una resolución baja del monitor/escritorio. Más tarde, cuando el usuario cambia a una resolución más alta del monitor/escritorio la reproducción va a fallar porque la posición de los elementos UI en la pantalla ha cambiado. Por lo tanto, en la medida que el ambiente del computador cambia la reproducción se hace cada vez menos confiable.

En conclusión, existe la necesidad para un sistema sencillo y amplio de grabación de macros y herramientas de grabación UI que sean compatibles con los lenguajes

macro existentes y grabadoras de macro específicas a una aplicación. Al mismo tiempo, existe la necesidad para un sistema amplio y sencillo de herramientas de grabación de macros y UI que trabajen con interfaces de usuario dinámicas.

Resumen

Acá se presentan técnicas y herramientas para grabar y reproducir actividades en la interfase gráfica de un usuario ("GUI") para varias aplicaciones en forma confiable. Por ejemplo, una herramienta de grabación de macros y UI simplifica la tarea de construir macros y grabar actividad GUI para algunas aplicaciones mientras apoya las grabadoras de macros específicas a una aplicación, y también apoya la grabación de macros y GUI dentro de una interfase de usuario dinámica.

De acuerdo con un primer aspecto, una herramienta recibe acciones internas de macro de una interfase gráfica de usuario basada en aplicaciones como fichas opacas. Por ejemplo, un programa de software graba la actividad de un usuario en su ambiente de grabación nativo. Basada en las entradas grabadas, el programa de software genera fichas opacas que se transmiten a la herramienta.

De acuerdo con un segundo aspecto, una herramienta combina fichas grabadas de una acción de macro interna con otra actividad de la interfase del usuario. Por ejemplo, un programa de software le pasa acciones de macro internas grabadas como una herramienta de ficha opaca. La herramienta combina esas fichas opacas con fichas recibidas de una grabadora de sistemas UI.

De acuerdo con un tercer aspecto, una herramienta recoge fichas grabadas asociadas con una actividad de la interfase de usuario de distintas aplicaciones y replica la actividad del usuario en la interfase suministrando cada ficha grabada a la aplicación con la que cada ficha se asocia. Por ejemplo, la actividad en la interfase de un usuario que abarca múltiples aplicaciones se graba como fichas grabadas y se combina con una herramienta en un código de reproducción. La

herramienta replica la actividad grabada pasando las fichas grabadas del código de reproducción al software del programa apropiado.

De acuerdo con un cuarto aspecto, una herramienta pasa dos fichas grabadas al programa de software para reproducirlas en la grabadora interna del programa de software. Por ejemplo, unas instrucciones de macro internas se extraen de un conjunto de fichas grabadas y se suministran a la grabadora interna del programa de software asociado con las fichas de reproducción.

Rasgos y ventajas adicionales del invento se harán aparentes de la siguiente descripción de implementaciones que aplican en referencia a los dibujos anexos.

Descripción breve de los Dibujos

La figura 1. es un diagrama de bloque de un marco de trabajo de una interfase de usuario grabando y reproduciendo.

La figura 2. es un diagrama de bloque de una herramienta de grabación UI que abarca todo el sistema.

La figura 3. es un diagrama de flujo que muestra una técnica para grabadora macros nativos a una aplicación con una herramienta de grabación UI que abarca a todo el sistema.

La figura 4. es un diagrama de bloque de una herramienta de reproducción UI que abarca a todo el sistema.

La figura 5. es un diagrama de flujo que muestra una técnica para administrar la reproducción de actividad de macros nativos a una aplicación con una herramienta de reproducción UI que abarca a todo el sistema.

La figura 6. es un diagrama de bloque de un ambiente de computación adecuado para la implementación de grabación y reproducción UI que abarca a todo el sistema.

Descripción Detallada

La siguiente descripción está dirigida a las técnicas y herramientas para grabación y reproducción de macros y UI para diversos softwares basados en GUI. Por ejemplo una herramienta de grabación de macros y UI que abarque todo el sistema utiliza una técnica de grabación apropiada dependiendo en el tipo de actividad de usuario que se esté grabando y las capacidades de grabación de la aplicación que estén utilizando (a veces referidas acá como grabaciones "inteligentes"). La herramienta puede utilizar un mecanismo de grabación basado en UIA para grabar la actividad del usuario contra una aplicación compatible con UIA. Si la aplicación no es compatible con UIA (o incluso si es, pero se desea grabar con otros mecanismos), la herramienta puede grabar los pasos simples (por ejemplo, clics con un botón del ratón, entradas del teclado) que el usuario utiliza en su interacción con una aplicación. La herramienta puede utilizar un mecanismo de grabación especial en varias circunstancias. Estas circunstancias incluyen la grabación de actividades descritas en un lenguaje de macro interno específico a una aplicación (grabado por la aplicación utilizando su propio mecanismo interno de grabación) y grabando las actividades contra una interfase dinámica de usuario. Las herramientas que se describen acá pueden grabar actividad contra varias clases de aplicaciones, software de sistemas operativos, y otro tipo de software.

Solicitud de Patente US Serial No. aa/bb/cc, llamada "herramientas de Grabación/Reproducción para aplicaciones basadas en UI", archivada en junio 15, 2004, cuya divulgación se incorporan acá por referencia, describe varias herramientas de grabación y reproducción. Algunas de estas herramientas utilizan un marco de trabajo UIA para grabar entradas basadas en el valor de identificación de un elemento UI ("UID"). UIA genera un ID compuesto que identifica el elemento UI en un GUI. El UID depende por lo menos en parte en la localización del elemento UI en la interfase del usuario y su relación con otros elementos UI. Cuando la herramienta utiliza un UID para grabar y reproducir actividades que

Involucran un elemento específico GUI, cambios en la localización del elemento pueden causar un error en la reproducción.

Por ejemplo, supóngase que un usuario reconfigura la interfase de usuario eliminando un botón en una barra de menú. Cuando una herramienta de reproducción intenta activar el botón que se ha eliminado, una falla en reproducción se origina. Otro ejemplo frecuente es cuando la aplicación se actualiza y ciertos rasgos se renombran para acomodar otros rasgos. Por ejemplo, supóngase que en una versión la función "Abrir Archivo" se mueve para acomodar una función "Archivo Nuevo", y "Abrir Archivo" se convierte en el cuarto ítem en la lista del menú "Archivo". Macros que estén basados en la interfase antigua pueden no funcionar porque el elemento UI se ha movido y/o el UID ha cambiado.

En algunas implementaciones, una herramienta de grabación de macros y UI que abarque todo el sistema utiliza grabación inteligente para resolver este tipo de problemas. Una aplicación graba las acciones de un usuario en lenguaje de macros interno de la aplicación. Luego de grabar, los pasos grabados se expresan en el lenguaje macro interno de la aplicación y se pasan como fichas de pasos grabados de la aplicación a la herramienta de grabación de macros y UI. La herramienta de grabación de macros y UI que abarca todo el sistema puede entonces encapsular las fichas de los pasos grabados y/o asociar las fichas de los pasos grabados con información de marcación. En algunas implementaciones, cada instrucción de macro se convierte en una ficha separada de grabación de pasos. Alternativamente, una serie de instrucciones relacionadas de lenguaje macro se envuelven en una ficha de un paso grabados. Esas fichas mantienen una secuencia temporal de las instrucciones en lenguaje macro subyacente que les permite reproducirse en la aplicación tal como fueron grabadas, mientras también permiten la interacción con la herramienta de grabación de macros y UI que abarca todo el sistema. Luego de recibir las fichas de los pasos grabados, la herramienta puede generar un código que refleja los pasos grabados. Luego el código puede ser

20

reproducido para recrear los pasos grabados o, alternativamente, el código se puede editar para probarse, o corregir fallas u otros propósitos.

Por ejemplo, supongo se que un procesador de palabras tiene unas instrucciones en lenguaje macro "Abrir (nombre de archivo)" que engancha el objeto modelo interno de la aplicación para abrir un archivo con nombre "nombre de archivo". Durante la grabación con la herramienta Inteligente de grabación, las instrucciones de macro se pasan a una ficha de grabación de un paso asociada con otra información apropiada, y almacenada para reproducirse, momento en el cual la instrucción subyacente, "Abrir (nombre de archivo)", se pasa a una ficha de un paso grabado de la herramienta Inteligente de grabación devuelta a la aplicación. Esto permite a la aplicación utilizar un su propio ambiente nativo de grabación para reproducir y grabar actividades UI dentro de un marco de trabajo de grabación y reproducción UI que abarque todo el sistema. Un macro que abarque todo el sistema se puede escribir de tal manera que no esté sujeto a problemas asociados con cambios en los menús o cambios en la Interfase de usuario o asociados con actualizaciones de software, siempre y cuando el lenguaje macro subyacente permanezca compatible con la versión anterior del software. Aún más, el marco de trabajo para la grabación que abarque todo el sistema provee una interfase de grabación y reproducción unificada. También simplifica la interpolación de macros y grabaciones UI separa distintas aplicaciones en procesos de grabación de macros y UI combinados.

Como parte del marco de trabajo de grabación UI que abarca todo el sistema, cuando una aplicación no utiliza la grabadora de macros interna u otra grabadora de UI específica a una aplicación, se puede utilizar un mecanismo de grabación distinto. Por ejemplo, se utiliza un sistema de grabación basado en UIA, como se describe en la solicitud de Patente de EU llamada "Herramientas de Grabación/Reproducción para Aplicaciones Basadas en UI", que trabajan bien para grabar actividad en la interfase del usuario contra aplicaciones que tienen un

41

sistema de menús estático. En forma alterna, si una aplicación no apoya las operaciones de acuerdo con la arquitectura de Automatización UI del marco de trabajo RPF para grabar UI a través de todo el sistema puede simplemente grabar los pasos directos ejecutados por un usuario en la medida que el o ella interactúan con la aplicación basada en UI. En algunos casos, varias técnicas de grabación se usan en combinación. Por ejemplo en algunos casos de prueba las entradas directas del usuario, los datos basados en UIA, y los datos de la grabadora interna de macros son todos grabados.

La grabación de pasos directos (por ejemplo, clics del ratón, entradas en el teclado) que un usuario ejecuta interactuando con aplicaciones basadas en UI es relativamente sencilla; el reto está en cómo representar y procesar ésta información cuando una aplicación tiene una interfase de usuario configurable (por ejemplo, una que permite que los menús y los items de menús se cambien a gusto), cuando una aplicación se actualiza y modifica la interfase del usuario, o cuando una aplicación procesa información de macros o grabación de UI en lenguaje interno. Conceptualmente, algunas técnicas y herramientas que se describen acá mejoran la grabación integrando las capacidades internas de grabación de la aplicación con una interfase abstracta y amigable del usuario a través de la cual el usuario puede convertir los pasos grabados y las entradas en un código de reproducción. Como se anotó anteriormente las aplicaciones con sus capacidades internas de grabación pueden invocar su grabadora interna para mapear las acciones de un usuario a los objetos internos de la aplicación, de esta manera resolviendo el "bloqueo de actualización" y problemas de reconfiguración de menús con una simple grabación menú/menú-item.

En algunas implementaciones, macros específicos a una aplicación o datos de grabación UI se recogen y se convierten en fichas de grabación de pasos utilizando una técnica de marcación. Las técnicas y las herramientas crean fichas opacas definiendo las interacciones del usuario con una aplicación. Una vez las fichas de

22

los pasos grabados se han generado, otras acciones grabadas (por ejemplo, eventos UIA que se encuentran en fichas) se puede convertir en un código de programación para reproducir en el mismo computador o en otro, potencialmente con fichas de pasos grabados por la aplicación todavía incrustados dentro del código de programación.

En algunas situaciones comunes, las herramientas y técnicas que se describen acá permiten a una herramienta de grabación de macros y UI que abarca todo el sistema ínter operar con varios tipos de programas de software (por ejemplo aplicaciones compatibles con UIA, aplicaciones con grabadoras internas de macros, aplicaciones que no son compatibles con UIA) y varios tipos de interfaces de usuario (por ejemplo, estáticas, dinámicas). Estas técnicas y herramientas hacen la grabación más confiable y robusta mediante el uso de las capacidades internas de grabación de la aplicación cuando son apropiadas. Aún más, las herramientas y técnicas proveen al usuario con una interfase de grabación de macros y UI que es consistente. Las técnicas y herramientas se pueden implementar de varias maneras, y se pueden utilizar combinadas o en forma separada.

I. Marco de Trabajo para la Grabación y Reproducción UI

Un marco de trabajo para la grabación y reproducción UI ("UI RPF") es un marco de trabajo extensible designado para facilitar la grabación y reproducción de actividad UI contra aplicaciones u otros programas de software. Módulos que se pueden incorporar dentro del marco de trabajo incluyen filtros, agregados, marcadores, y generadores de código. La figura 1. muestra un UI RPF 100 en donde varias de las técnicas descritas acá se pueden implementar. Básicamente, un UI RPF 100 graba (o trabaja con otro software que graba) lo que un usuario está haciendo y reproduce (o trabaja con otro software que reproduce) sus acciones en el mismo computador o en otro. GUI RPF 100 es un marco de trabajo

23

23

(por ejemplo, a nivel de sistema) a través de varias aplicaciones que se puede utilizar con aplicaciones múltiples y en muchas distintas configuraciones, y que está diseñado para facilitar la grabación y reproducción de distintos tipos de actividad del usuario. Es un fundamento en el cual las herramientas de grabación y reproducción se pueden construir. El UI RPF 100 puede utilizar servicios de software del sistema de arquitectura GUI 120, tal como la Automatización UI de Microsoft, para facilitar la grabación UI pero éste no es un requisito del marco de trabajo 100.

La figura 1. muestra varios módulos asociados con el UI RPF 100. El GUI RPF 100 interactúa con el software de arquitectura del sistema UI 120 para convertir los datos UI de entrada grabados tales como clics del ratón, entradas en el teclado, y entradas de las aplicaciones 125, 126 en datos que son más significativos para la grabación y reproducción UI. El marco de trabajo 100 recibe los pasos grabados de aplicaciones tales como un editor de texto 125 una calculadora 126 a través del software de arquitectura del sistema GUI 120, que posiblemente pasa como una ficha que encapsula las acciones de un usuario en la Interfase del sistema. En algunas implementaciones el GUI RPF 100 utiliza un mecanismo de grabación y reproducción de GUI descrito en la solicitud de patente U.S. llamada "Herramientas para Grabación/Reproducción de Aplicaciones Basadas en UI", para grabar y reproducir actividad UI recibida a través del software de arquitectura del sistema GUI 120. En forma alternativa, el UI RPF 100 utiliza un mecanismo distinto de grabación y reproducción UI.

Los módulos del UI RPF 100 también incluyen el módulo de grabación y reproducción Inteligente 110 para operar con distintas aplicaciones que tienen su grabadora de macros nativa. Sin tener en cuenta el software de arquitectura del sistema UI 120, el marco de trabajo 100 recibe los pasos grabados y otra información de macros en fichas de pasos grabadas en una aplicación tal como un procesador de palabras 115, una hoja de cálculo 116 un editor de fotos 117. Las

aplicaciones 115, 116, 117 tienen sus capacidades de grabación internas para mapear las entradas de un usuario a acciones en el ambiente de grabación nativo de la aplicación, cuando las aplicaciones 115, 116, 117 pasan las acciones grabadas internamente como fichas al marco de trabajo 100 a través del módulo Inteligente de grabación y reproducción UI 100.

El UI RPF 100 mantiene una lista de las aplicaciones para los cuales actividad de macros o UI se pueden grabar. Una aplicación nueva que es vista por el UI RPF 100 se consulta por lo menos la primera vez que la aplicación se ejecuta, para ver si es compatible con interacciones a través de una interfase inteligente de grabación. Si la aplicación no es compatible con interacciones a través de la interfase inteligente, entonces las acciones del usuario se pueden grabar de acuerdo con las capacidades de grabación propias de la aplicación y devueltas a través de la interfase. En forma alterna, las acciones de un usuario se pueden grabar a través del mecanismo interno de grabación de la aplicación (con datos que pasan a través de la Interfase inteligente de grabación) y a través del software de arquitectura del sistema UI 120, donde ambas opciones son posibles. O, incluso si ambas opciones son posibles, un usuario, una aplicación, o una herramientas que funcionan a través de todo el sistema pueden decidir si utilizan el mecanismo interno de grabación de la aplicación o grabación a través del software de arquitectura del sistema UI 120, pero no ambos. O el usuario, la aplicación, o la herramienta que funciona a través de todo el sistema pueden decidir grabar pasos directos (por ejemplo, movimientos del ratón, clics en los botones del ratón, entradas en el teclado) asociadas con las acciones del usuario, ya sea en combinación o en forma separada con otros datos grabados.

Cuando la actividad de un usuario se graba a través del mecanismo de grabación interno de la aplicación a su vez se graba a través de la Interfase inteligente con el UI RPF, la aplicación convierte la actividad del usuario grabada en fichas. El contenido de estas fichas se define en aplicaciones de Visual Basic ("VBA") o en

25

otro lenguaje o módulo de objetos utilizado por la aplicación. Últimamente, la información guardada en esta "carga útil" de las fichas no la interpreta el GUI RPF, pero únicamente las aplicaciones respectivas para la cual la información se encuentra en su formato nativo. El UI RPF 100 puede convertir en fichas otras fichas de pasos grabados y asociar los datos con otros datos relevantes tales como el nombre de la aplicación.

Con la herramienta que abarca a todo el sistema, los datos de fichas de pasos grabados se pueden concatenar con fichas de pasos grabados obtenidos de otras aplicaciones con una grabadora nativa, o con datos grabados obtenidos a través del software de arquitectura del sistema GUI 120, de tal manera que se pueda implementar la grabación y reproducción a nivel del sistema.

Adicionalmente a grabar macros y datos UI, el marco de trabajo 100 puede producir códigos que reproducen la actividad grabada y/o encapsulan datos que permiten que otro software reproduzca la actividad grabada.

Por ejemplo, en el ambiente de Microsoft Windows, un usuario comienza la grabación de actividad UI, luego hace clic en el botón de "Inicio", y abre un procesador de palabras que trabaja con la Interfase Inteligente de grabación. En el contexto de UI RPF, las acciones de un usuario incluyendo los clics del ratón en el botón de "Inicio" se graban. Cuando el procesador de palabras se abre, el UI RPF chequea para ver si el procesador de palabras es compatible con una interfase inteligente de grabación. Si lo es, el UI RPF invoca la herramienta apropiada dentro del procesador de palabras mismo para grabar las acciones ejecutadas. Si el procesador de palabras puede trabajar con la arquitectura del sistema UI, la herramienta también puede grabar las acciones del usuario (eventos de Automatización UI) en conjunto con el software de arquitectura del sistema UI. Adicionalmente, la herramienta también puede grabar los pasos directos (por ejemplo, clics con el botón del ratón, entradas en el teclado) asociadas por las acciones de un usuario, que pueden ser particularmente relevantes si el

75

procesador de palabras no trabaja con la interfase inteligente de grabación y no está habilitado para trabajar con el software de arquitectura del sistema UI. Por lo tanto, el UI RPF puede utilizar mecanismos alternos para grabar reproducir.

Este marco de trabajos 100 puede incluir elementos de software y/o hardware. La relación que se muestra entre los módulos en la figura 1 indica el flujo principal de información; otras relaciones no se muestran por simplicidad. Dependiendo de la implementación, los módulos se pueden añadir, omitir, romper en múltiples módulos, combinar con otros módulos y/o remplazarse con módulos parecidos. En forma alternativa, el marco de trabajo con módulos diferentes y/o otras configuraciones de módulos ejecutan una o más de las técnicas de grabación y reproducción UI descritas en el presente documento.

II. Grabación Inteligente

La figura 2 muestra una herramienta de grabación UI 220 que abarca todo el sistema en conjunto con las técnicas varias descritas acá con los cuales se puede implementar. El UI RPF de la figura 1 incluye la herramienta de grabación 220. En forma alternativa, la herramienta de grabación 220 se usa como parte de otro marco de trabajo UI.

La herramienta de grabación 220 abarca todo un sistema en el sentido de que graba información de la arquitectura del sistema UI para grabar UI con ciertas aplicaciones, pero también para trabajar con aplicaciones que graban actividad en la interfase del usuario en su propio ambiente de grabación de macros nativo. Estas aplicaciones de grabación nativas intercambian datos de grabación de macros o UI con la herramienta 220 a través de una interfase especial de grabación nativa (que se muestra como un RPF API en la figura 2). Esto permite que la herramienta 220 utilice (o interopere con) las capacidades de grabación propias de la aplicación, que permiten que la grabación UI para un usuario

dinámico de la interfase y/o grabación UI que se puede reproducir con cambios en la interfase del usuario por cambios de versión.

Esta herramienta de grabación 220 controla el mecanismo que se va usar para grabar macros o datos UI. Por ejemplo, dependiendo de las entradas de un usuario, la configuración del usuario, u otros criterios, la herramienta 220 decide si utiliza la grabación nativa de la aplicación, grabación del sistema UI o ambas para una aplicación particular. La herramienta también puede grabar entradas directas UI del ratón del usuario y del teclado, en forma separada o en combinación con otros mecanismos de grabación. La herramienta 220 también puede grabar datos de macros UI para cada una de múltiples aplicaciones que estén en uso, o para aplicaciones seleccionadas entre aplicaciones múltiples que estén en uso. Por ejemplo, si un usuario configura la herramienta de grabación 220 para que grave únicamente datos de un procesador de palabras, mientras que un usuario graba cortar y pegar de una hoja de cálculo a un procesador de palabras, únicamente los pasos relacionados con el procesador de palabras se graban.

A. Ruta de Grabación Nativa

Para grabaciones nativas de macros o UI, un usuario puede interactuar con la aplicación 240 utilizando dispositivos de entrada tales como un ratón, un teclado, una almohadilla táctil, una pantalla sensible, una pantalla Braille, una palanca de mando, un micrófono, u otros dispositivos similares. En la medida que el usuario interactúa con la aplicación 240, las entradas del usuario se procesan y son grabadas por la grabadora de macros/UI 245 como pasos en el ambiente nativo de grabación de macros de la aplicación, como se expresa en un lenguaje como VBA, PerfectScript, M4, TeX o cualquier otro lenguaje macro. Las capacidades internas de grabación de la aplicación proveen un marco de trabajo robusto para adquirir datos que se puede reproducir la aplicación en forma predecible y confiable. Los pasos grabados sólo están limitados por las limitaciones en el ambiente nativo de

20

grabación de la aplicación. Por ejemplo, si una aplicación le permite al usuario declarar variables en un macro, esas variables se pueden grabar y convertir en fichas.

La aplicación 240 convierte en fichas los pasos grabados (que se muestran como la marcación 247) de tal manera que se le pueden pasar a la herramienta 220. Luego, la aplicación pasa las fichas de los pasos grabados a la herramienta 220 a través de la interfase. Las fichas le son opacas a la herramienta 220, lo que significa que la herramienta 220 puede manipular las fichas de los pasos grabados como bloques de datos opacos y discretos cuyo contenido no lo interpreta la herramienta 220. La aplicación 240, por el contrario, puede interpretar y reaccionar al contenido de las fichas de los pasos grabados. El contenido está en el formato nativo de la aplicación 240. Por ejemplo, la aplicación 240 encapsula como una ficha grabada las instrucciones macro para abrir un archivo exactamente como fue grabado (por ejemplo, como datos binarios, comandos de texto, o instrucciones DLL).

La aplicación 240 convierte cada paso grabado y discreto en una ficha, o la aplicación 240 agrupa múltiples pasos grabados en una sola ficha. En cuanto al mecanismo de marcación, la aplicación 240 puede agrupar datos de pasos grabados en uno o más campos de una estructura de datos para la ficha, que luego se pasa como una copia o una referencia a la herramienta 220. La aplicación 240 puede pasar los datos de los pasos grabados como parámetros de un método enunciado, que la herramienta 220 recibe y manipula como datos opacos. La aplicación 240 puede utilizar una hoja de reglas XML (que define una serie de reglas para convertir en fichas los datos de los pasos grabados) para reformatear o reorganizar los datos de los pasos grabados en fichas. O, la aplicación 240 utiliza otro mecanismo de conversión en fichas. Conceptualmente, la conversión en fichas se puede ver como la organización de los pasos grabados en una estructura de datos de fichas, o como envolver los pasos grabados en información de fichas.

21

29

Esto "esconde" el lenguaje de instrucciones macro de la herramienta, creando una ficha que es independiente del lenguaje macro pero cuyo contenido se puede reconocer en el ambiente de grabación nativo apropiado.

La aplicación 240 puede anexar y añadir datos de marcación a los pasos grabados cuando está convirtiendo los datos en fichas. La marcación de datos provee detalles pertinentes a la reproducción, por ejemplo, información de la aplicación (por ejemplo, el nombre de la aplicación y su versión), una marca de tiempo para indicar el orden, Información de privilegios (por ejemplo, que derechos se necesitan para reproducir los pasos), etc.

La herramienta 220 recibe las fichas (que se muestra como el receptor de fichas 235) que vienen de una o más aplicaciones a través de la interfase especial de grabación nativa. Mientras la herramienta 220 no interpreta el contenido de las fichas de los pasos grabados, puede (en vez de o adicional a la aplicación) añadir capas de datos a las fichas de los pasos grabados y/o asociar las fichas con información de marcación tal como el nombre de la aplicación, marcaciones de tiempo, valores de índice, etc. de tal manera que los pasos se pueden reproducir con la aplicación correcta y en la secuencia apropiada. En este sentido, se considera que la herramienta 220 también hacen la función de conversión a fichas.

La aplicación 240 es compatible con una interfase nativa de grabación en el sentido de que formatea la información para pasarla a la herramienta 220 a través de la interfase, hace las llamadas apropiadas a través de la interfase, implementa métodos de respuesta para la herramienta 220, etc. La herramienta 220 es compatible con una interfase en el sentido de que implementa métodos para que la aplicación llame, manipule información que viene de otras aplicaciones, inicie métodos de respuesta de la aplicación, etc. El mecanismo exacto para llamar y procesar utilizado por la interfase depende de la implementación.

30

B. Ruta de Grabación del Sistema UI

La aplicación 240 con la grabadora de macros nativa y/o otra aplicación 260 pueden ser compatibles con las interacciones con la herramienta de grabación del sistema 220 a través del software de arquitectura del sistema UI 120. Para grabación con la arquitectura del sistema UI, varios componentes 225 de la herramienta de grabación 220 recolectan (a) datos de entrada del usuario, (b) actividad UI en el monitor (como cambios de foco, cambios de selección, creación de ventanas etc.), (c) información de pistas acerca del ambiente, como aplicaciones que están corriendo, versiones de OS, información de localización, lenguaje UI por defecto, lenguaje UI actual, dispositivos de hardware y configuraciones, modalidad de seguridad, y/o otro tipo de datos que pueden ayudar a describir y recrear el ambiente apropiado para la reproducción, (d) resolver los datos de entrada del usuario en elementos en la interfase del usuario afectados por la entrada del usuario, y (e) expresar la actividad del usuario como eventos lógicos grabados. Estos eventos lógicos grabados ayudan a proveer salidas de UI grabadas y legibles y reproducciones de grabaciones UI confiables. La grabación lógica de eventos se puede considerar como un tipo de fichas de grabación UI, pero a diferencia de las fichas de los pasos grabados opacos con datos de grabación nativos, la herramienta 220 interpreta la grabación lógica de dichos eventos para visualizar, editar, convertir un código, etc.

En algunas implementaciones, la herramienta 220 ejecuta una grabación con la arquitectura del sistema UI con filtros y agregados como se describe en la Solicitud de Patente de E. U. llamada "Herramientas de Grabación/Reproducción para Aplicaciones basadas en UI". De otra manera, la herramienta 220 utiliza otra técnica para la grabación con la arquitectura del sistema UI.

La herramienta 220 puede desplegar los pasos grabados con la arquitectura del sistema UI en un visor tal como una ventana de solución de problemas, un visor de códigos fuente, o un editor de texto de tal manera que los usuarios pueden leer

31

y buscar errores en los datos. En forma típica, la herramienta 220 no muestra información acerca de las fichas de los pasos grabados con datos de grabación nativos, en la medida que el contenido de estas fichas es opaco a la herramienta 220. Cuando las fichas de los pasos grabados se asocian con información de marcación tal como el nombre de la aplicación, marcaciones de tiempo, etiquetas de acciones u otras anotaciones hechas por el usuario o la aplicación, la herramienta 220 puede mostrar dicha información. Los datos se pueden almacenar en un archivo de registro para análisis futuro.

C. Otras Entradas de Usuario

Los componentes varios 225 de la herramienta de grabación 220 pueden también recoger directamente datos de entrada del usuario y monitorear en forma similar la actividad UI, seguir información acerca del ambiente, tal como aplicaciones que estén corriendo, etc. grabar estos eventos ayuda en situaciones donde una aplicación no es compatible con la interfase inteligente de grabación o la arquitectura del sistema UI. Estas acciones grabadas también se pueden mostrar al usuario en un visor de tal manera que pueda corregir errores en los datos. En forma típica, los datos de las acciones directas del usuario son menos legibles y confiables para reproducción que otro tipo de fichas grabadas como se explicó anteriormente en los antecedentes. Sin embargo, cuando las otras técnicas de grabación son inaplicables, esto provee un mecanismo para grabar datos de aplicación no nativos y no UIA en el UI RPF que abarca todo el sistema. Los datos que se graban de las acciones directas del usuario se pueden almacenar en forma similar en un archivo del registro para análisis futuro.

3/2

D. Código de Reproducción

En algunas implementaciones, la herramienta 220 combina fichas de diferentes aplicaciones y modalidades de grabación en un único archivo mediante el intercalado de datos. Por ejemplo, fichas de pasos grabados para acciones grabadas en forma nativa para distintas grabadoras nativas se combinan en un solo flujo de entrada, junto con las fichas para las acciones en la arquitectura del sistema UI y otras entradas del usuario. En forma típica, las entradas de un usuario se reciben en forma serial y la secuencia de fichas mantiene de esta manera el orden en el flujo. Marcaciones de tiempo, valores de índice, o información de la aplicación, puede marcar las fichas para ayudar a mantener el orden apropiado. O, la herramienta 220 crea archivos separados para distintas aplicaciones y modalidades de grabación.

Para fichas de acciones del sistema UI, la herramienta 220 utiliza un generador de códigos 230 para generar un código de reproducción 250 a partir de las fichas. Esto se puede hacer automáticamente como cuando el usuario selecciona la opción de "Generar Código" en la herramienta 220. El código de reproducción 250 se puede generar en una variedad de formatos, incluyendo C#, XML, C++, JAVA, o cualquier otro lenguaje de programación, macros, guiones, u otros comandos o instrucciones para una interfase de reproducción u otros mecanismos para manejar aplicaciones basadas en UI. Por ejemplo, el generador de código 230 mapea las fichas de acciones del sistema UI con instrucciones correspondientes en el código de reproducción 250. Para más detalles sobre la generación de códigos en un ambiente de arquitectura de sistema UI, ver la solicitud de patente de E.U. llamada, "Herramientas de Grabación/Reproducción para Aplicaciones Basadas en UI", entregada el 15 de junio de 2004.

La herramienta 220 opcionalmente pone las fichas de los pasos grabados para acciones grabadas en forma nativa dentro del código de reproducción 250 tal

33

como un libreto de archivo que mantiene las fichas de los pasos grabados en la secuencia temporal correcta. O, la herramienta 220 utiliza un generador de código 230 para generar el código de reproducción 250 que incluye las fichas de los pasos grabados. Es de anotar, que las fichas de los pasos grabados que encapsulan datos de grabación nativos son opacas a la herramienta 220, por lo tanto estas fichas pasan en forma típica a través del código de reproducción 250, y no son interpretadas por la herramienta 220 para conversión a otras expresiones.

El código de reproducción 250 se puede visualizar, guardar, editar, modificar, añadir a, eliminar de, compilar, o utilizar antes de la reproducción, con las limitaciones esperadas en la medida en que la herramienta puede visualizar, modificar, etc. fichas opacas de pasos grabados. Poder editar el código de reproducción 250 le da a los usuarios un mecanismo conveniente y poderoso para crear macros, generar casos de prueba automáticos, automatizar pasos en un programa, etc. el código generado se puede editar para añadir expresiones condicionales, para cambiar variables, para añadir regresiones, y otras instrucciones de programación al código. Por ejemplo, cuando se va a reproducir, suponga que una pieza del código de reproducción altera un archivo de sistema y requiere al usuario para que se conecte con el Administrador de derechos. De ésta forma, una expresión condicional se añade al código de reproducción, que chequea los derechos apropiados antes de correr una parte del código. En otro ejemplo, supóngase que diseñadores de software quieren probar un programa repitiendo la misma serie de instrucciones una y otra vez, de tal manera, que "una orden circular" se añade al código de reproducción, que hace que el código de reproducción se repita cierto número de veces.

El código de reproducción 250 se puede entonces utilizar para administrar la reproducción de la actividad grabada tal como se describe más abajo. El código 250 se puede reproducir en el mismo computador o en un computador diferente.

34

E. La Herramienta de Arquitectura y su Uso

En general, los módulos varios de la herramienta 220 se pueden utilizar como un plug-in, lo que significa que se pueden remplazar con otros módulos que tengan una funcionalidad similar para distintas aplicaciones. Por ejemplo, el generador de código 230 en la herramienta 220 se puede utilizar como un plug-in, de tal manera que un diseñador puede rehusar partes de la herramienta 220 y cambiar la forma en que el código se genera cambiando el generador de códigos. La herramienta 220 por lo tanto es extensible; su funcionalidad se puede cambiar para distintas aplicaciones. En algunas Implementaciones, la herramienta 220 presenta un menú de opciones para módulos plug-in, permitiéndole al usuario controlar la carga de módulos apropiados para la funcionalidad deseada por usuario.

Por ejemplo, en modalidad de grabación "pesada", la herramienta 220 recoge información detallada acerca de las actividades UI y registra la información en forma comprensiva. Esto permite una reproducción más confiable pero es posible que perturbe la ejecución del software en el cual se graba la actividad. O, en una modalidad de grabación "ligera", la herramienta 220 recoge menos información y, por lo tanto, es menos probable que perturbe la ejecución del software donde se graba la actividad. Sin embargo, esto puede dar suficiente información para la reproducción en el mismo ambiente, para mostrar la actividad del usuario y apoyar al personal, o para otros propósitos.

En forma similar, la herramienta 220 puede grabar información en una modalidad interactiva en donde los elementos UI se seleccionan con un rectángulo que los cobija o mostrando en forma intermitente lo que se ha grabado. Otro tipo de retroalimentación que se muestra en forma intermitente se puede utilizar para mostrar al usuario la aplicación para la cual datos UI se están grabando. O, en modalidad interactiva, se le pide al usuario entrar información de anotación descriptiva para cada acción o acciones que la herramienta 220 o la aplicación 240

no puedan resolver. De otra manera, la herramienta 220 puede operar en el trasfondo para grabar actividad UI sin perturbar al usuario.

La herramienta 220 es responsable de cargar los otros módulos plug-in en la medida que se vayan necesitando (por ejemplo módulos que recogen datos, que convierten datos en fichas, que generan códigos, etc.), imponiendo seguridad (por ejemplo, chequeando los privilegios), desovando trayectorias grabadas, sincronizando actividades entre los módulos, construyendo y manteniendo consultas que se utilizan en la grabación, registrando errores (por ejemplo, a un archivo local u opcionalmente a una red de servicios o un servidor), y recogiendo información acerca de las grabaciones (por ejemplo, versión OS, localización de información, configuración de hardware, etc.).

La relación que se muestra entre los módulos en la figura 2 indica el flujo principal de información; las otras relaciones no se muestran por simplicidad. Dependiendo de la implementación, los módulos se pueden añadir, omitir, dividir en módulos múltiples, combinar con otros módulos, y/o reemplazar con otros módulos parecidos. De otra manera, una herramienta con distintos módulos y/o otras configuración desde módulos ejecutan una o más de las técnicas de grabación descritas acá.

La interfase del usuario de la herramienta 220 puede ser una interfase a la manera VCR que estaba dispuesta alrededor de la ventana de grabación, anclada a la parte superior o inferior de la pantalla del computador, o minimizada en la barra de tareas. La interfase incluye botones para actividades tales como "Grabar", "Pausa", "Detener", "Reproducir", como también un área para presentar descripciones de los pasos grabados cuando tal descripción está disponible. Para ver ejemplos de tales interfases, de la solicitud de Patente E.U. "Herramientas para Grabación/Reproducción en aplicaciones basadas en UI".

26

F. Técnicas de Grabación, Cronometraje de las Grabaciones

Existen dos modalidades para transferir las fichas a la herramienta 220 durante la grabación: modalidad en lotes y modalidad Interactiva. En general, en la modalidad en lotes, las fichas de pasos grabadas se recogen en la aplicación y se entregan a la herramienta 220 cuando la solicita. Esta solicitud de recoger las fichas puede ocurrir cuando el foco ya no está en la aplicación que graba, y también puede ocurrir periódicamente para distribuir la carga de grabación.

En modalidad Interactiva, en contraste, las fichas se envían de la aplicación que se graba a la herramienta 220 tan pronto ocurren. Una desventaja de enviar las fichas en forma Interactiva es que aumenta la carga de procesamiento a través en la CPU. En una situación particular, ya sea una de las modalidades interactiva o en lotes es preferible a la otra, dependiendo en la aplicación que se estaba grabando, la cantidad de datos grabados, y que tan rápido se están generando las fichas de los pasos grabados.

La figura 3 muestra una técnica 300 para grabar actividad del macro nativa a una aplicación con una herramienta de grabación UI que abarca todo el sistema. Una herramienta y una aplicación tal como la herramienta de grabación 220 y la aplicación 240 que se muestran en la figura 2 ejecutan la técnica 300. De otra manera, otra herramienta y/o aplicación ejecuta la técnica 300.

La herramienta chequea 310 para ver si la aplicación utiliza una interfase Inteligente de grabación nativa. Algunas aplicaciones no son compatibles con las Interacciones a través de una interfase. Otras aplicaciones pueden ser compatibles con las Interacciones a través de una interfase, pero el usuario o la herramienta determinan no utilizar la grabación nativa. Por ejemplo, cuando un usuario primero inicia una aplicación la herramienta se crea para ver si la aplicación es compatible con Interacciones a través de la Interfase Inteligente de grabación nativa, y si lo es, añade el nombre de la aplicación a la lista de aplicaciones que son compatibles con Interacciones a través de la Interfase. O, la herramienta chequea para ver si la

37

aplicación es compatible con la interfase inteligente de grabación nativa en el momento en el cual se inicia la grabación. Si una aplicación no es compatible con la interacción a través de la interfase, la herramienta puede añadir la aplicación a la lista de aplicaciones que no son compatibles.

Si la interfase de grabación nativa inteligente no se utiliza, la herramienta utiliza otra grabadora 315 para grabar actividad UI. Por ejemplo, la herramienta utiliza una grabadora de software UIA para aplicaciones capaces de UIA, o de otra manera, una grabadora de entradas UI directas para aplicaciones no capaces de UIA. Mientras que la figura 3 muestra las rutas alternativas de grabación, la herramienta puede utilizar ambas, la grabación nativa y la otra grabación simultáneamente.

Cuando la interfase de grabación inteligente nativa se utiliza, la aplicación recibe una entrada del usuario y graba datos macro 320. De esta manera, la grabadora nativa de la aplicación graba los pasos ejecutados por el usuario contra la aplicación.

El proceso de grabar datos macro continúa hasta que la aplicación, con la herramienta, determina 325 que no debe continuar. Este chequeo puede evaluar una variedad de condiciones incluyendo si el usuario ha terminado en forma selectiva el proceso de grabación, si la herramienta ha solicitado fichas de la aplicación en modalidad por lotes, si un contador o timer ha vencido en la modalidad interactiva o por lotes, si hay suministros para convertir en fichas, si un elemento UI designado se ha contactado, si se ha alcanzado un número específico de pasos grabados, etc.

La aplicación luego convierte en fichas 330 los datos de macro grabados, produciendo una o mas fichas que son ocultas a la herramienta, por ejemplo, la aplicación utiliza un mecanismo de conversión a fichas como se describió anteriormente y/o añade información de marcación como se describió

anteriormente. De otra manera, la herramienta ejecuta algunas o todas las conversiones a fichas o adiciones de información.

La herramienta o la aplicación evalúan 335 si continúan con la grabación de datos de macro. Una vez más, este chequeo puede evaluar una de varias condiciones, incluyendo si el usuario ha terminado en forma selectiva el proceso de grabación, si la conversión a fichas ha sido exitosa o si se debe solicitar información de anotación del usuario, etc.

La herramienta opcionalmente convierte 340 las fichas que recibe a un código tal como un guión en el cual las fichas están incrustadas. Por ejemplo, la herramienta utiliza un mecanismo de generación de código como se describió anteriormente.

De otra manera, las etapas varias de la técnica 300 se ejecutan en forma separada o combinada con otras etapas. Por ejemplo, mientras la figura 3 muestra la conversión a fichas 330 y 340 que ocurre luego de la grabación de datos de macro 320, en forma alternativa una o ambas de estas operaciones se ejecutan en forma concurrente con la grabación 320.

III. Reproducción Inteligente

La figura 4 muestra una herramienta de reproducción 420 UI que abarca todo el sistema en conjunto con otras técnicas varias descritas acá que se pueden implementar. El UI RPF de la figura 1 incluye la herramienta de reproducción 420. En forma alternativa, la herramienta de reproducción 420 se utiliza como parte de otro marco de trabajo UI.

La herramienta de reproducción reproduce la actividad UI recibiendo códigos de reproducción y conduciendo a la aplicación a través del sistema de servicios UI, pasando fichas a la aplicación para que se interpreten en su propio ambiente nativo, o a través de reproducir los pasos directos asociados con la actividad UI. La herramienta de reproducción 420 abarca todo el sistema en el sentido de que manejan la información de reproducción de la arquitectura del sistema UI para

reproducción de grabaciones UI con ciertas aplicaciones, pero también inter opera con aplicaciones que reproducen la actividad en la interfase del usuario en su propio ambiente de grabación de macros nativo. Estas aplicaciones de grabación nativas intercambian datos de grabación de macros o UI con la herramienta 420 a través de una interfase especial de reproducción nativa (que se muestra como el RPF API Inteligente en la figura 4). Esto permite que la herramienta 420 controle (y opere con) las capacidades de grabación propias de la aplicación, que pueden permitir la reproducción UI para Interfases dinámicas de usuario y/o reproducción UI para interfases que han cambiado con distintas versiones de una aplicación.

La herramienta de reproducción 420 controla las rutas que se han de utilizar cuando se reproducen macros o datos UI. Por ejemplo, dependiendo de que datos se han grabado, la herramienta 420 decide si debe utilizar la reproducción de la grabadora nativa de la aplicación, reproducción en el sistema UI, o ambas. De otra manera, la herramienta también puede reproducir los pasos grabados directamente. La herramienta 420 puede reproducir datos de macros o UI simultáneamente para cada aplicación que está en uso, o para cada aplicación seleccionada entre múltiples aplicaciones en uso.

A. Lector

La herramienta de reproducción 420 recibe el código de reproducción 450 como una entrada de un generador de códigos tal como el generador de códigos 230 que se muestra en la figura 2 y en otras partes. El código de reproducción 450 puede estar en una variedad de formatos, tales como C#, XML, C++, JAVA, o cualquier otro lenguaje de programación, guiones, macro u orientado hacia los documentos, u otros comandos o instrucciones, para una interfase de reproducción u otros mecanismos para administrar aplicaciones basadas en UI para acciones en el sistema UI. Fichas de pasos grabados para acciones de grabación nativas se

40
pueden incluir en el código 450 tal como un archivo de guión (que se puede dirigir directamente a la ficha que pasa 435) o incrustarse en otro código 450.

En algunas implementaciones, la herramienta de reproducción 420 recibe un único archivo que combina códigos y fichas de distintas aplicaciones y modalidades de grabación intercalando los datos. Por ejemplo, fichas de pasos grabados para acciones de grabación nativas para distintas grabadoras nativas se combinan en un solo flujo, junto con un código para acciones en la arquitectura del sistema UI y otras entradas de usuario grabadas. O, la herramienta 420 recibe archivos separados para distintas aplicaciones y modalidades de grabación.

En la herramienta de grabación 420, el lector 410 recibe el código de reproducción 450 y verifica que el código 450 esté correcto. Por ejemplo, la herramienta de reproducción recibe un archivo que contiene un código de reproducción e intenta verificar que el archivo esté internamente consistente y consistente con el status actual del sistema. Si se encuentran algunas inconsistencias, se notifica al usuario, y, en algunas implementaciones, se le pide al usuario que complete manualmente los pasos. De otra manera, la reproducción se termina.

Para reproducción de acuerdo con la ruta de la arquitectura del sistema UI, el lector a 410 le pasa el código 450 al reproductor de la arquitectura del sistema UI 425 para control programático de la aplicación como se especifica en el código 450. Para reproducción de acuerdo con la ruta de grabación, el lector 410 le pasa las fichas de los pasos grabados para grabación de datos nativa a la ficha pasante 435.

B. Ruta de Reproducción de la Grabadora Nativa

Para reproducción con la grabadora nativa de macros o UI, la herramienta de reproducción 420 pasa fichas (que se muestran como la ficha pasante 435) de la herramienta 420 a la aplicación 440 que incluye el ambiente de grabación nativo apropiado. La herramienta 420 le pasa la ficha a la interfase de reproducción

nativa especial. Antes de hacer esto, la herramienta 420 puede interpretar capas de datos adicionales que fueron añadidas a las fichas de los pasos grabados, incluyendo información de marcación tal como el nombre de la aplicación, marcas de tiempo, valores de índice, etc., de tal manera que los pasos se reproducen en la aplicación correcta y en la secuencia apropiada.

A diferencia de otros tipos de información en el código 450, las fichas de los pasos grabados en forma nativa no se interpretan directamente con la herramienta de reproducción 420. En cambio, las fichas se pasan como bloques de datos opacos por la ficha pasante 435 a la aplicación 440 (que se muestra como el componente que acepta las fichas 442). La aplicación 440 interpreta las fichas, pasando los datos de grabación de macros de la fichas al reproductor de macros 445. En la aplicación 440, los datos de macros grabados se interpretan en el ambiente nativo de grabación de macros. Los datos de macros se expresan en un lenguaje tal como VBA, PerfectScript, M4, TeX u otro lenguaje macro. Las capacidades internas de grabación de una aplicación suministran un marco de trabajo robusto y confiable para la reproducción, a veces permitiendo una reproducción que no es posible a través de la arquitectura de sistema UI.

La aplicación 440 extrae los pasos grabados preparándolos para la reproducción. La aplicación 440 puede extraer un paso grabado de una solo ficha, o la aplicación 440 puede extraer múltiples pasos de acciones grabadas de una solo ficha. En cuanto a la mecánica de la conversión a fichas en orden inverso, la aplicación 440 puede leer datos de acciones de pasos grabados de uno o más campos de la estructura de datos para la ficha, que pasó de la herramienta 420 como una copia por referencia. O, la aplicación 440 puede desempacar datos de acciones de pasos grabados que pasa como parámetros de una llamada de método. O, la aplicación 440 puede utilizar otro mecanismo para extraer las acciones de los pasos grabados como fichas. Conceptualmente, la extracción puede verse como una lectura de los datos de las acciones de los pasos grabados de una estructura de datos en una

42
ficha, o desempacando los datos de acciones de pasos grabados de la información de la ficha. Esto "revela" las instrucciones del lenguaje macro, que son reconocibles en el ambiente nativo de grabación apropiado.

La aplicación 440 puede proceder entonces a marcar los datos anexos o añadidos a los datos de las acciones de pasos grabados cuando se generan las fichas de los datos. La marcación de los datos suministra detalles pertinentes a la reproducción, por ejemplo, información de la aplicación (por ejemplo el nombre de la aplicación y su versión), una marca de tiempo para indicar el orden, información sobre privilegios (por ejemplo, que derechos se tienen para reproducir los pasos), etc.

La herramienta 420 es compatible con una interfase especial de reproducción nativa en el sentido de que formatea la información para pasarla a la aplicación 440 a través de la interfase, hace las llamadas apropiadas a través de la interfase, implementa métodos para regresar las llamadas para la aplicación 440, etc. La aplicación 440 es compatible con una interfase en el sentido de que implementa métodos para que la herramienta 420 llame, manipule información que le ha sido pasada por la herramienta, invoca los métodos para regresar las llamadas de la herramienta 420, etc. El mecanismo de procesamiento y de llamadas utilizado por la interfase dependen de la implementación.

C. Ruta de Reproducción del Sistema UI

La aplicación 440 con su grabadora reproductora de macros nativos y/o otras aplicación 460 pueden apoyar la reproducción de acuerdo a la arquitectura del sistema UI con el reproductor de la arquitectura del sistema UI 425. El reproductor 425 es en esencia un motor para controlar en forma programática rasgos de una aplicación basada en UI. Para esta ruta de reproducción, la herramienta de reproducción 420 procesa el código 450 con el reproductor 425 para reproducir los pasos grabados y la actividad UI a través de Interacciones programáticas con la aplicación basada en UI.

43
En algunas implementaciones, la herramienta 420 ejecuta una reproducción en la arquitectura del sistema UI con componentes tal como se describe en la solicitud de patente de E.U. llamada "Herramientas de Grabación/Reproducción para Aplicaciones basadas en UI", entregada el 15 de Junio de 2004. En forma alterna, la herramienta 420 utiliza otra técnica para reproducir con la arquitectura del sistema a UI.

Al inicio de la reproducción, una comparación se podría hacer entre las condiciones existentes y las condiciones bajo los cuales la grabación UI se hizo. Si se detectan diferencias significativas, se le informa al usuario, y se lo ofrece la opción de continuar.

En algunos casos, el código de reproducción se ejecuta por sí mismo y no requiere entradas o acciones de control adicionales. En forma alternativa, las acciones del sistema UI se ponen en una lista de espera para reproducción, que le permite al módulo de reproducción 425 procesar cada paso grabado en orden. El reproductor 425 obtiene un paso grabado de la lista, espera que la aplicación adecuada esté disponible, espera que el proceso esté listo para las entradas, ejecuta los pasos grabados, reporta el resultado de los pasos a la herramienta de reproducción 420 o en algún otro lado. Si cualquiera de esos pasos falla, una función simple de dormir se invoca, el reproductor 425 simplemente espera un período de tiempo determinado antes de intentar reproducir los pasos grabados de nuevo. O, cuando un paso no se puede ejecutar, se notifica al usuario y se le pide que complete paso. Esto permite que fallas en la reproducción se puedan manejar sin que interrumpan a la herramienta de reproducción o cualquier otra aplicación asociada. En vez de depender en el reproductor 425, algunas aplicaciones lo pueden sustituir con su propia herramienta de reproducción para correr el código generado.

D. Arquitectura de Herramientas y Usos

En general, los módulos varios de la herramienta 420 funcionan como un plug-in, lo que significa que pueden ser reemplazados con otros módulos con una

funcionalidad similar para distintas aplicaciones. La herramienta 420 por lo tanto es extensible; su funcionalidad se puede alterar para distintas aplicaciones. La herramienta 420 presenta un menú de opciones para módulos que funcionan como un plug-in, permitiendo de esta manera al usuario controlar la carga de módulos apropiados para la funcionalidad deseada.

La herramienta de reproducción suministra varias opciones que se pueden configurar. La herramienta de reproducción 420 puede configurarse para indicar como se debe ejecutar la reproducción. Por ejemplo, la herramienta de reproducción se puede configurar para que pause durante la reproducción cuando un botón de "Pausa" se activa. O, la herramienta 420 pausa cuando se requiere la entrada de un usuario para continuar procesando el código de reproducción. La herramienta de reproducción se puede configurar para ejecutar reproducciones tan rápido como el sistema de computación corriendo el código pueda "Pintar". En forma alternativa, la reproducción ocurre tan rápido como es posible sin esperar a que se complete la Pintura. En algunas implementaciones, series múltiples de reproducción se corren en forma simultánea contra aplicaciones separadas.

Para mejorar la seguridad de la reproducción, la herramienta de grabación 220 de la figura 2 puede codificar el código de reproducción y/o las fichas incrustadas en él. Si es el caso, la herramienta de reproducción 420 codifica el código y/o las fichas. Otras medidas de seguridad incluyen el requerimiento de un nombre y una contraseña que el usuario debe ingresar para la reproducción, no registrar estos datos, chequear la modalidad de seguridad del código de reproducción contra el modo de seguridad del usuario (por ejemplo, requiriendo el status del administrador al usuarios que trate de reproducir el guión a nivel del administrador), y únicamente permitiendo reproducción almacenada (reproducción de un código y/o fichas que no se encuentran en un archivo) cuando se puede verificar que el contenido del buffer lo grabó el usuario actual, en la sesión actual.

La herramienta 420 es responsable por cargar otros módulos plug-in en la medida que se necesiten, chequeando la seguridad (por ejemplo, chequeando privilegios), desovando secuencias de reproducción, sincronizando actividades entre módulos, construyendo y manteniendo colas utilizadas para reproducción, registrando errores (por ejemplo, a un archivo local u opcionalmente a un servidor de red o servicios), y recolectando información acerca de la reproducción.

La relación que se muestra entre módulos en la figura 4 indica el flujo principal de información; otras relaciones no se muestran por simplicidad. Dependiendo en la implementación, los módulos se pueden añadir, omitir, dividir en múltiples módulos, combinar con otros módulos, y/o reemplazar con módulos parecidos. En forma alternativa, una herramienta con distintos módulos y/o otras configuraciones de módulos ejecutan una o más de las técnicas de reproducción descritas acá.

La interfase de usuario de la herramienta 420 puede ser una interfase del estilo VCR localizada alrededor de la ventana de reproducción, anclada en la parte superior o inferior de la pantalla del computador, o minimizada en la barra de tareas. La interfase incluye botones para acciones como "Pausa", "Detener", y "Reproducir", como también un área para presentar descripciones de los pasos que se reproducen cuando esta descripción está disponible. Como ejemplo de tal interfase, ver la solicitud de patente de E.U. llamada "Herramientas de Grabación/Reproducción para Aplicaciones basadas en UI".

E. Técnicas de Reproducción

La figura 5. muestra una técnica 500 para administrar la reproducción de actividad macro nativa a la aplicación con una herramienta de reproducción UI que abarca todo el sistema. Una herramienta y aplicación tal como la herramienta de reproducción 420 y la aplicación 440 que se muestran en la figura 4 ejecutan la técnica 500. En forma alternativa, otra herramienta y/o aplicación ejecutan la técnica 500.

La herramienta recibe el código 510 con las fichas de los pasos grabados. Por ejemplo el código es un archivo de guión que incluye fichas. O, el código es un código en lenguaje de programación que tiene fichas incrustadas.

La herramienta luego verifica 520 el código. La herramienta puede verificar si las fichas son correctas y tienen el orden y la consistencia interna, o verifica que la herramienta y la aplicación estén listas para una reproducción correcta. Haciendo esto, la herramienta puede considerar marcar la Información añadida a las fichas para verificar la identificación con las secuencias. Por ejemplo, la herramienta utiliza el mecanismo de verificación descrito anteriormente. Si una acción grabada no se puede verificar, se le pide al usuario que termine la reproducción o, alternativamente, ejecute una acción.

La herramienta luego pasa las fichas del paso grabado 540 a la aplicación apropiada. La aplicación procesa 550 las fichas, extrayendo datos de macros grabados, y reproduciendo las acciones de macro grabadas en el ambiente nativo de grabación y reproducción de la aplicación para reproducir los pasos ejecutados por un usuario contra la aplicación. Procesar las acciones grabadas en forma nativa incluye la reproducción de la actividad UI grabadas tan suave como sea posible con muy poca o ninguna interferencia por el usuario. Así, una herramienta de grabación del sistema UI puede efectivamente administrar e interactuar con el ambiente de grabación interno de la aplicación con un alto grado de confiabilidad.

La herramienta o aplicación determina 560 si continúa la reproducción. Por ejemplo, la herramienta y la aplicación terminan cuando el código con las fichas se han terminado de reproducir, cuando el usuario termina la reproducción, o cuando la reproducción termina debido a un error, o la herramienta y aplicación esperan cuando la reproducción se detiene para solicitar entradas de datos del usuario.

De otra manera, varias etapas de la técnica 500 se ejecutan en forma separada con varias combinaciones en conjunto con otras etapas. Por ejemplo, mientras que la figura 5. muestra la verificación 520 y el paso 540 ocurriendo antes del

procesamiento de las fichas 550, alternativamente una o ambas de estas operaciones se ejecutan en forma concurrente con el proceso 550 para fichas en línea de reproducción.

IV. Ambiente de Computación

Las herramientas descritas anteriormente para grabación y reproducción de macros y UI para software diverso basado en GUI (por ejemplo, la herramienta 200 de la figura 2) y macros y técnicas de grabación UI se pueden implementar en una variedad de dispositivos de cómputo y ambientes, incluyendo computadores de varias formas (personal, estaciones de trabajo, servidores, de mano, laptops, tabletas u otros móviles), redes de computación distribuidas, servicios Web, como algunos ejemplos generales. La herramienta y técnicas de macro y grabación UI se pueden implementar en un circuito de hardware, como también sin grabaciones de macro, software de procesamiento y visualización 980 ejecutándose dentro de un computador u otros ambientes de computación, tales como los que se muestran en la figura 6.

La figura 6. muestra un ejemplo generalizado de un ambiente de computación adecuado 600 en donde las técnicas descritas se pueden implementar. El ambiente de computación 600 no pretende sugerir ninguna limitación en cuanto al alcance ni funcionalidad de las herramientas y técnicas acá descritas, pues ellas se pueden implementar en diversos ambientes de computación de propósito general o específico.

Con referencia a la figura 6, el ambiente de computación 600 incluye por lo menos una unidad de procesamiento 610 y una memoria 620. En la figura 6, esta configuración básica 630 se incluye con una línea punteada. La unidad de procesamiento 610 ejecutan instrucciones ejecutables por computador y puede ser un procesador real o virtual. En un sistema de multi-procesamiento, las unidades de multi-procesamiento ejecutan instrucciones ejecutables por computador para

aumentar el poder de procesamiento. La memoria 620 puede ser una memoria volátil (por ejemplo, registros, cache, RAM), o memoria no volátil (por ejemplo, CD ROM, EEPROM, memoria flash, etc.) o alguna combinación de las dos. La memoria 620 almacena software 680 que implementa las herramientas y técnicas de grabación inteligentes.

Un ambiente de computación puede tener rasgos adicionales. Por ejemplo, el ambiente de computación 600 incluye el almacenaje 640, uno o más dispositivos de entrada 650, uno o más dispositivos de salida 660, y una o más conexiones de comunicación 670. Un mecanismo de interconexión (que no se muestran) tal como un bus, un controlador, o una red que interconecta los componentes del ambiente de computación 600. En forma típica, el software del sistema que opera (no se muestran) suministra un ambiente operativo para que otros software se ejecuten en el ambiente de computación 600, y coordine las actividades de los componentes en el ambiente de computación 600.

El almacenaje 640 puede ser removible o no removible, e incluye discos magnéticos, cintas magnéticas o casetes, CD-ROMs, CD-RWs, DVDs, o cualquier otro medio que se pueda utilizar para almacenar información y que se pueda acceder dentro del ambiente de computación 600. El almacenaje 640 almacena instrucciones para la grabación macro o UI y el software de reproducción 680.

El dispositivo(s) 650 (por ejemplo, para dispositivos que funcionan como un punto de control en la arquitectura de conectividad de los dispositivos) puede ser un dispositivo táctil de entrada como un teclado, un ratón, un bolígrafo, un trackball, un dispositivo de entrada de voz, un escáner, u otros dispositivos que puedan proveer entradas al ambiente de computación 600. Para audio, el dispositivo(s) de entrada 650 puede ser una tarjeta de sonido o dispositivo similar que acepte entradas de audio en forma analógica o digital, o un lector CD-ROM que suministra pruebas de audio al ambiente de computación. El dispositivo(s) 660 puede ser un

monitor, impresora, parlantes, escritor de CDs u otros dispositivo que provea salidas al ambiente de computación 600.

Las conexiones de comunicación 670 permiten la comunicación a través de un medio de comunicación con otra entidad de computación. El medio de comunicación suministra información tal como instrucciones ejecutables por computador, audio/vídeo u otros medios de información, u otros datos en señal de datos modulada. Una señal de datos modulada es una señal que tiene una o más de sus características fijas o alteradas de tal manera que codifica la información en la señal. Como ejemplo, y no como limitante, un medio de comunicación incluye técnicas alambradas e inalámbricas implementadas con transmisores tales como eléctricos, ópticos, infrarrojos, RF, acústicos y demás.

Las técnicas de macro y grabación y reproducción UI que se muestran acá pueden describirse en el contexto general de medios legibles por computador. Medios legibles por computador son cualquier medio disponible que se pueda acceder dentro de un ambiente de computación. Como ejemplo, y no como limitante, en el ambiente de computación 600, los medios legibles por computador incluyen la memoria 620, el almacenaje 640, los medios de comunicación, y una combinación de cualquiera de los antedichos.

Las técnicas presentadas se pueden describir en el contexto general de instrucciones ejecutables por computador, tales como las incluidas en módulos de programas, que se ejecutan en un ambiente de computación en un procesador real o virtual. Generalmente, los módulos de programa incluyen rutinas, programas, bibliotecas, objetos, clases, componentes, estructuras de datos, etc. que ejecutan distintas tareas o implementa tipos de datos abstractos. La funcionalidad de los módulos de programa se puede combinar o dividir entre módulos de programa como se desee en las varias implementaciones. Instrucciones ejecutables por computado para módulos de programa se pueden ejecutar dentro de un ambiente de computación local o distribuido.

5

Para claridad en la presentación, la descripción detallada utiliza términos como "determinar", "generar", "ajustar", y "aplicar" para describir operaciones de computador en un ambiente de computación. Éstos términos son abstracciones de alto nivel para operaciones ejecutadas por un computador, y no se deben confundir con actos ejecutados por un ser humano. Las operaciones de computador actuales correspondientes a estos términos varían dependiendo de la implementación.

Como existen muchas implementaciones posibles en las cuales los principios de mi invento se pueden aplicar, yo reivindicó dichas implementaciones como mi invento en la medida que entren en el alcance y el espíritu de las siguientes reivindicaciones.

7

Reivindicamos:

1. Un medio legible por computador que almacena instrucciones legibles por computador para hacer que un sistema de computación programado consecuentemente ejecuta un método que consta de:

recibir una pluralidad de acciones internas de macro que pasan en forma opaca como una o más fichas de pasos grabados de una aplicación basada en una interfase gráfica de usuario, y grabando uno o más pasos opacos grabados como fichas con un grabador externo UI.

2. El medio legible por computador de la reivindicación 1 donde una pluralidad de acciones de macro internas se generan por una grabadora interna de macros de la aplicación y donde una o más fichas de pasos opacos grabados las puede manipular una grabadora externa UI que interopera con la grabadora interna de macros de la aplicación.

3. El medio legible por computador de la reivindicación 1 comprende además:

recibir una pluralidad de acciones de sistema en la Interfase del usuario para Interacciones del usuario contra una segunda aplicación basada en una interfase gráfica de usuario;

generar una o más fichas de sistema de la interfase de usuario para una pluralidad de acciones de sistema en la interfase del usuario; y

Grabar una o más fichas de sistema en la interfase del usuario con una grabadora externa a UI.

4. El medio legible por computador de la reivindicación 1 incluye además: recibir una pluralidad de acciones directas del usuario para Interacciones del usuario contra una segunda aplicación basada en una interfase gráfica de usuario;

grabando la pluralidad de acciones directas del usuario con una grabadora externa UI

52

5. El medio legible por computador de la reivindicación 1 comprende además la asociación de una o más fichas opacas de pasos grabados con datos relevantes en la grabadora externa UI.
6. El medio legible por computador de la reivindicación 5 donde los datos relevantes incluyen información de identificación de la aplicación para la aplicación.
7. El medio legible por computador de la reivindicación 1 donde una o más fichas opacas de pasos grabados son recibidas por una grabadora externa de una aplicación UI en modalidad de lotes.
8. El medio legible por computador de la reivindicación 1 donde una o más fichas opacas de pasos grabados son recibidas por una grabadora externa de una aplicación UI en modalidad interactiva.
9. El medio legible por computador de la reivindicación 1 donde la grabadora externa UI inter opera con múltiples grabadoras internas de macro para distintas aplicaciones, donde las múltiples aplicaciones incluyen aplicaciones basadas en interfaces gráficas de usuario.
10. El medio legible por computador de la reivindicación 1 donde el método incluye además:
 - pasar una o más de las fichas opacas de pasos grabados de la grabadora externa UI devuelta a la aplicación; y
 - reproducir la pluralidad de acciones internas de macro.
11. Un medio legible por computador que almacena instrucciones ejecutables por computador para hacer que un sistema de computación programado concordantemente ejecute un método que comprende:
 - recibir un conjunto de una o más acciones expresadas en un primer lenguaje de grabación utilizado por un programa de software; y
 - representar el conjunto de una o más acciones como un conjunto de una o más fichas de grabación para un programa de sistema de grabación UI, donde una o mas fichas de grabación encapsulan una o más acciones de tal manera

53



que la una o más acciones son opacas dentro de la una o más fichas grabadas.

12. El medio legible por computador de la reivindicación 11 donde el programa de software genera una serie de una o más fichas de grabación.

13. El medio legible por computador de la reivindicación 11 donde el programa de grabación UI del sistema genera una serie de una o más fichas de grabación.

14. El medio legible por computador de la reivindicación 11 donde el método comprende además:

recibir un segundo conjunto de una o más acciones expresadas en un segundo lenguaje de grabación utilizado por un segundo programa de software; y
representar la segunda serie de una o más acciones como una segunda serie de una o más fichas de grabación para el programa de grabación UI del sistema.

15. El medio legible por computador de la reivindicación 11 donde el método comprende además:

recibir un segundo conjunto de una o más acciones expresadas en el lenguaje de grabación UI del sistema; y
representar el segundo conjunto de una o más acciones como una segunda serie de una o más fichas de grabación para el programa de grabación UI del sistema.

16. El medio legible por computador de la reivindicación 11 donde una o más fichas de grabación son recibidas por el programa de grabación UI del sistema en modalidad de lotes del programa de software.

17. El medio legible por computador de la reivindicación 11 donde una o más fichas de grabación son recibidas por el programa de grabación UI del sistema en modalidad interactiva del programa de software.

18. Una herramienta de grabación UI que abarca todo el sistema para grabar y reproducir actividad en la interfase del usuario, herramienta que consta de:

91

un módulo para recolectar un primer conjunto de datos asociados con la actividad en la interfase del usuario contra una pluralidad de distintas aplicaciones, donde el primer conjunto de datos se expresa en una pluralidad de lenguajes de grabación nativos para una pluralidad de aplicaciones diferentes;

un módulo para manipular por lo menos parte del primer conjunto de datos recogidos como un conjunto de fichas; y

un módulo para administrar la reproducción de la actividad en la interfase del usuario suministrando fichas a la pluralidad de aplicaciones diferentes con los cuales el primer conjunto de datos se asocia.

19. Las herramientas de la reivindicación 18 comprenden además un módulo para convertir el conjunto de fichas en un código de salida.

20. Las herramientas de la reivindicación 18 donde los módulos son extensibles.

21. Las herramientas de la reivindicación 18 comprenden además:

un módulo para recoger un segundo conjunto de datos asociado con una segunda actividad de Interfase de usuario, donde el segundo conjunto de datos se expresa en un lenguaje de sistema de grabación UI no nativo; y

un módulo para administrar la reproducción de la segunda actividad en la interfase del usuario reproduciendo un segundo conjunto de datos.

22. La herramienta de la reivindicación 18 donde el primer conjunto de datos comprende instrucciones en lenguaje macro.

23. La herramientas de la reivindicación 18 donde la recolección del primer conjunto de datos es en modalidad de lotes.

24. La herramienta de la reivindicación 18 donde la recolección del primer conjunto de datos es en modalidad interactiva.

25. Un medio legible por computador que almacena instrucciones ejecutables por computador para hacer que un sistema de computación programado de acuerdo ejecute un método que comprende: pasar una serie de una o más fichas de

ss

5

grabación de un programa de grabación UI de sistema a un programa de software para reproducción por una grabadora interna del programa de software; y extraer una serie de una o más acciones de grabación internas de una serie de una o más fichas de grabación, donde la serie de una o más acciones de grabación internas se expresan en un lenguaje de grabación interno utilizado por el programa de software, donde el programa de grabación UI del sistema inter opera con la grabadora interna para el programa de software.

26. El medio legible por computador de la reivindicación 25 donde el método comprende además:

pasar una segunda serie de una o más fichas de grabación del programa de grabación UI del sistema a un segundo programa de software para reproducir por una segunda grabadora interna de un segundo programa de software; y extraer una segunda serie de una o más acciones de grabación internas de una segunda serie de una o más fichas de grabación, donde la segunda serie de una o más acciones de grabación internas se expresan en un segundo lenguaje interno de grabación utilizado por un segundo programa de software, donde el programa de grabación UI del sistema interopera con una segunda grabadora interna para el segundo programa de software.

27. El medio legible por computador de la reivindicación 25 donde el método comprende además:

reproducir una segunda serie de una o más fichas de grabación para controlar programáticamente un segundo programa de software, donde una segunda serie de una o más fichas de grabación se expresan en lenguaje de grabación UI del sistema, donde el programa de grabación UI del sistema inter opera con un segundo programa de software.

86.

50

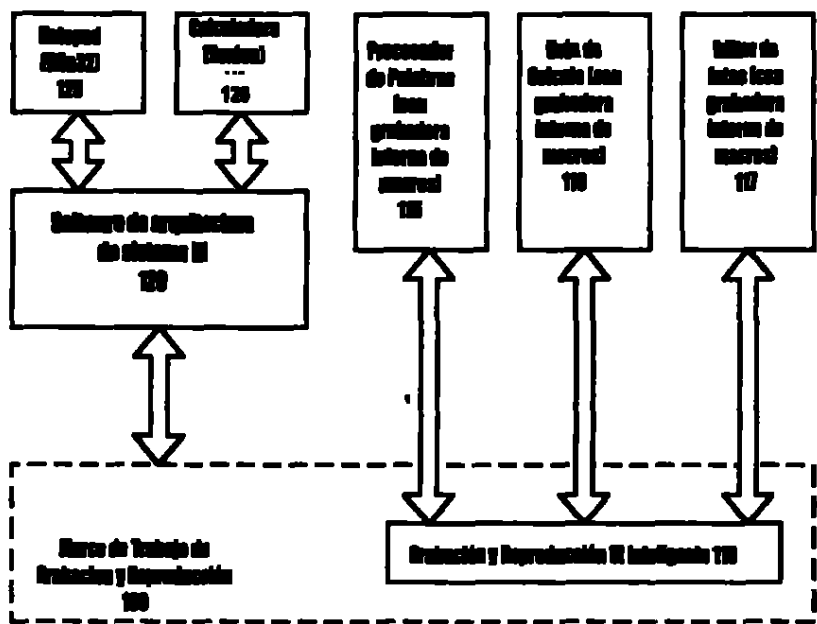
Resumen

Se describen técnicas y herramientas para grabar y reproducir en varias aplicaciones y en forma confiable actividad en la interfase gráfica del usuario ("GUI"), incluyendo aplicaciones que tienen una interfase de usuario dinámica. Las técnicas y herramientas se pueden utilizar en combinación o en forma separada. Por ejemplo, una herramienta de grabación recibe datos internos de macro de una aplicación basada en GUI como fichas opacas que están incrustadas en el código de reproducción. Una herramienta de reproducción ejecuta el código de reproducción pasando las fichas incrustadas devuelta a la aplicación original para reproducirse en la grabadora interna de la aplicación.

51

57

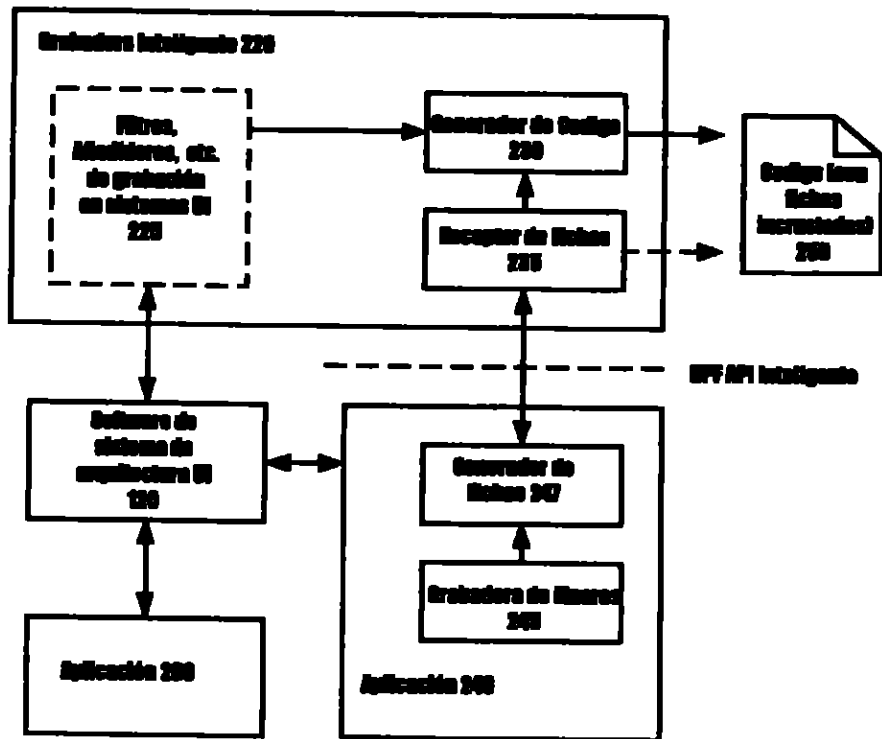
Figura 1



48

58

Figura 2



87

Figura 3

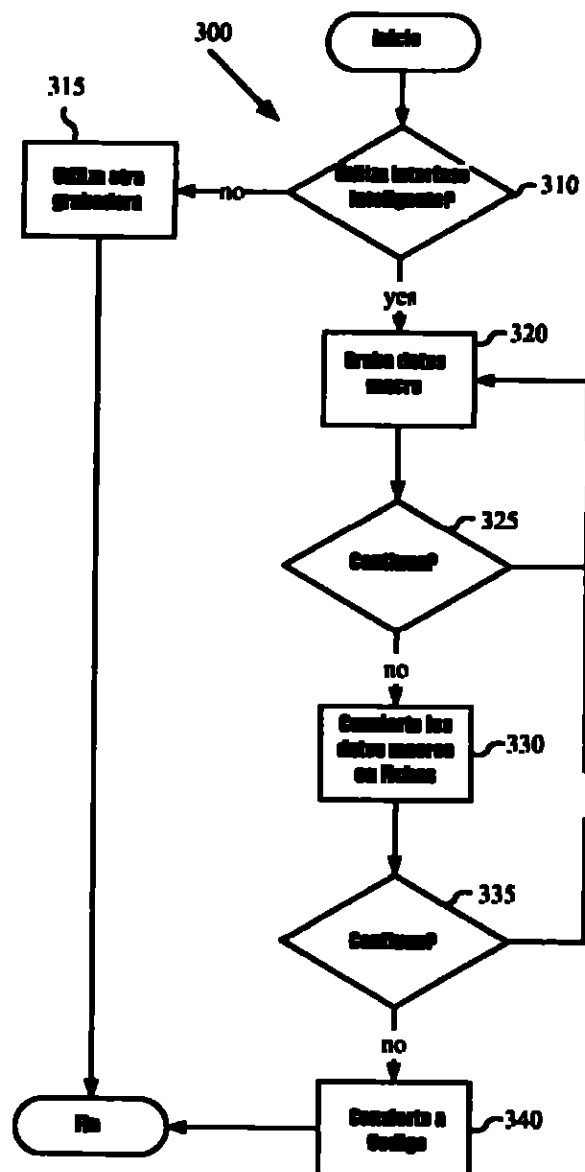


Figura 4

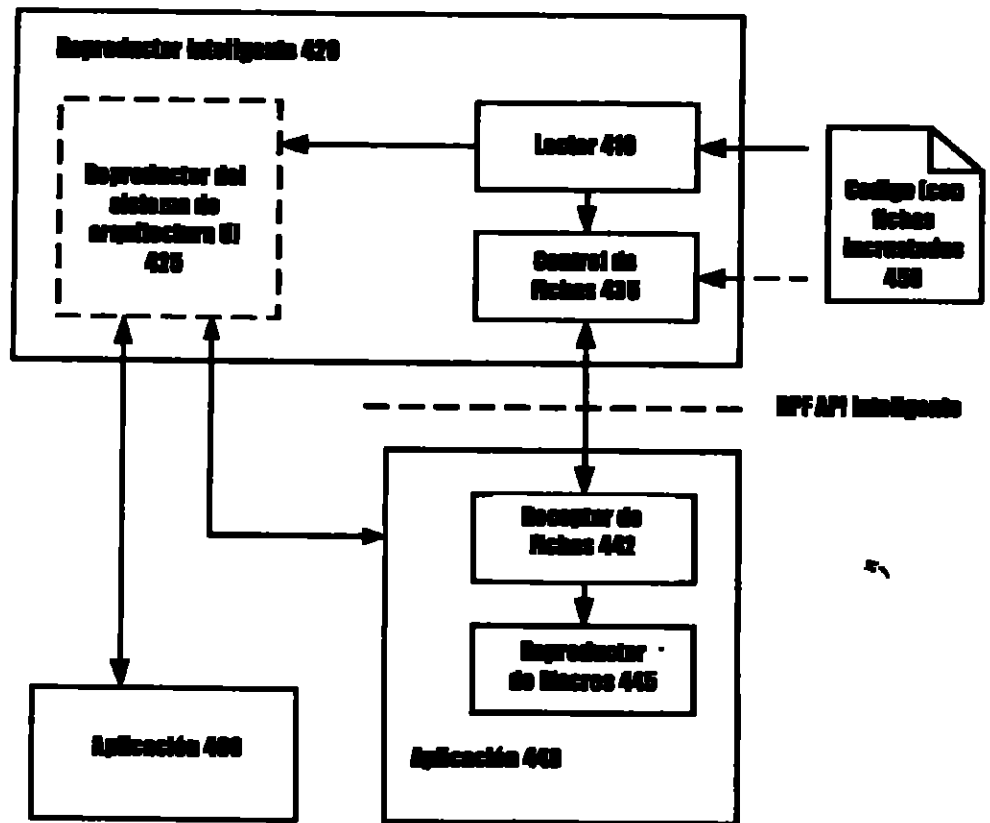
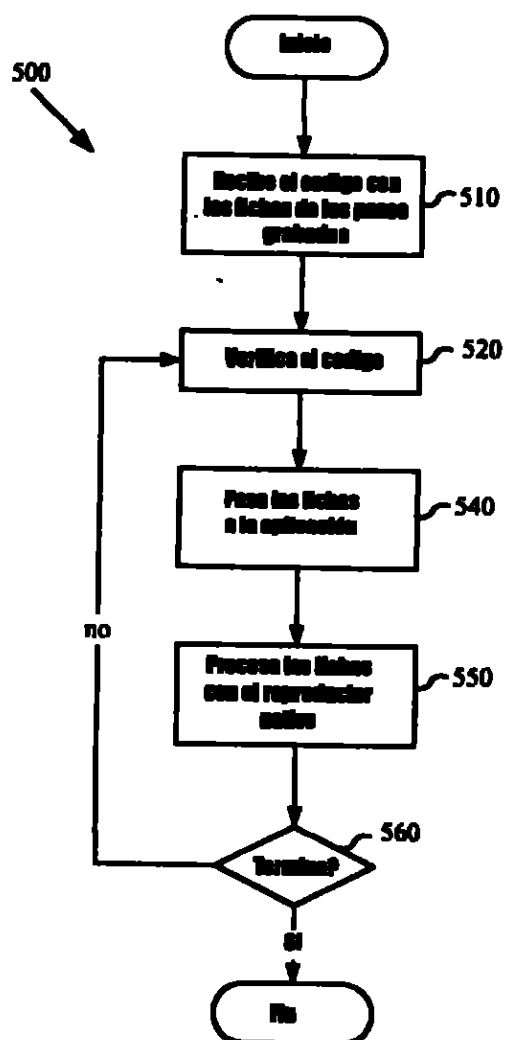
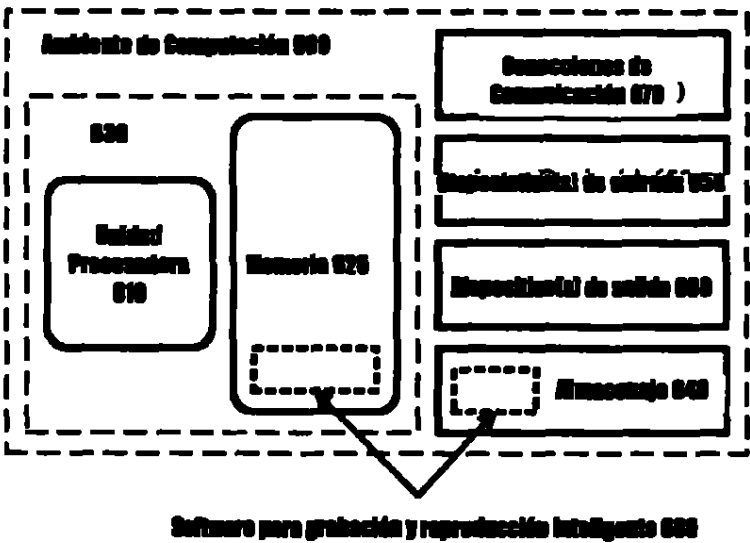


Figura 5



PR

Figura 6



63

Smart UI Recording and Playback Framework

Technical Field

Tools and techniques are described for recording user activity within various graphical user interface based applications. For example, a system-wide recording tool supports application-native macro or UI recording as well as non-native macro or UI recording.

Background

I. Graphical User Interface

On many modern computer systems, users interact with software programs through a graphical user interface ("GUI"). Basically, a GUI is an interface between computer and user that uses pictures rather than just words to solicit user input and present the output of a program. The typical GUI is made up of user interface elements ("UI elements"), which are those aspects of a computer system or program that are seen, heard, or otherwise perceived or interacted with by a user. For example, UI elements include items such as icons, buttons, dialog boxes, edit boxes, list boxes, combo boxes, scroll bars, pick lists, pushbuttons, radio buttons and various components of World Wide Web pages (e.g., hyper-links and images). In a typical computer program it is common to encounter literally thousands of UI elements.

Although an individual element of a GUI may appear to the user as a single item, it may actually consist of a number of separate items or sub-elements that have been combined together. For example, a toolbar item may consist of a list element, a combo box element, a scroll bar element, etc. Furthermore, each of these sub-elements themselves may be composed from other sub-elements. In this manner, UI elements can serve as building blocks for building other, more complex, UI elements. Such an approach is useful because the software managing the user interface can re-use the definitions of certain common elements when assembling them into composite elements.

Many UI elements in a GUI environment represent features of a program and are displayed on the computer screen so users can interact with the program by selecting,

64

highlighting, accessing, and operating them. This user interaction is done by maneuvering a pointer on the screen (typically controlled by a mouse or keyboard) and pressing or clicking buttons while the pointer is pointing to a UI element. For example, in a word processor, a user can maneuver the mouse to select an item on the program's menu bar, to click an icon from the tool bar, or to highlight blocks of text in the viewer window. Similarly, a user can use keyboard input to interact with a computer application. For instance, in the word processing program, a user can press "ALT-F," "CTRL-B," or other predefined keystroke combinations to access program features. Based on the input from the mouse or keyboard, the computer adds, changes, and manipulates what is displayed on the screen. GUI technologies provide convenient, user-friendly environments for users to interact with computer systems.

II. UI Automation

UI Automation ("UIA") is an accessibility framework for Microsoft Windows intended to address the needs of assistive technology products and automated test frameworks by providing programmatic access to information about the user interface ("UI"). For example, UIA allows a screen reader program to access information about the UI of a word processor, providing the reader program with the information it needs to provide audible cues to a visually impaired user. Through an application programming interface ("API") set of methods, UIA provides a well-structured mechanism for creating and interacting with a UI. Control and application developers use the UIA API set to make their products more accessible to different users through existing or new software (potentially written by other people) to access program menus and other UI elements. For example, braille screens, screen readers (narrators), magnifiers, and other software in Microsoft Windows can use UIA to facilitate computer use for users who otherwise may not have access.

In practice, UI Automation uses a hierarchy of UI elements located in a tree structure to provide reliable UI information to the operating system and computer applications. Elements of a GUI can be considered as nested within each other in order to accurately describe their organization. For example, at the very top of a tree structure is a desktop element that is representative of the GUI's "desktop" or default background

65

area. The desktop element has within it several application elements for application programs that have been invoked and that are ready to execute according to a user's instructions (e.g., a typical Microsoft Windows desktop may have several instances of applications such as Microsoft Word, Microsoft Excel, etc. loaded and ready to execute).

- 5 At a lower level in the tree structure hierarchy are frames associated with an application (e.g., a word processor application may have several frames visible to a user at any given time). Within each of the frames may be several documents, each document containing within it several UI elements (e.g., buttons, listboxes, etc.). UI elements may themselves be composites of other UI elements. For example, a dialog box or a combo box contains
- 10 other UI elements such as a button control. Furthermore, the button element may contain yet other UI elements. Such nesting can be arbitrarily deeper and include an arbitrary number of branches depending on the user interface and its component elements.

- For some operating system platforms, an instance of a UI element is assigned an identifier to help distinguish that particular UI element from other UI elements. For
- 15 example, in a Microsoft Windows based operating system, applications are associated with module identifiers that identify applications within a given desktop context. Also, some user interface platforms (e.g., Microsoft Windows, Swing for Java) use a numeric identifier (control ID) for certain UI elements. In some computing environments, such as a Microsoft Windows environment, UI elements are often associated with a class name
- 20 associated with the control class to which they belong. For instance, in a Microsoft Windows based system, common UI elements such as combo box, list box, and button are associated with class names such as ComboBox class, ListBox class, and Button class, respectively. Similarly, other UI frameworks may have names for their respective classes of UI elements.

- 25 Notably, these techniques identify a UI element's object class or type, but do not singularly provide a strong identifier that uniquely identifies a UI element across a reboot of the computer running the program, across a different build of the program when still in development, across the opening of an another instance of the same program, or for opening of the same program on another computer.

- 30 UIA overcomes these deficiencies by generating a composite ID that uniquely identifies a UI element in a GUI tree. UIA generates the composite identifier by adding

66

identification information (e.g., control name or control type) that is directly associated with a UI element to hierarchical identification information (e.g., parent control, child control, and/or sibling controls) and control pattern-specific information (e.g., depth of the UI element in the tree). For example, an identifier for a target UI element may be generated by collecting identifying information related to parent UI elements that describe the hierarchical arrangement between a target leaf UI element and the root element at desktop. Through the concept of a path, the related identifiers for a UI element's unique hierarchy and parentage can be leveraged to identify it uniquely and persistently.

The unique identifier (persistent ID) provides easy access to individual UI elements so that the functionality of a program hosting UI elements can be programmed and tested, and so that a particular UI element can be identified to other program modules. For additional information about UIA, see, for example, the documentation available through the Microsoft Developer Network.

III. Macros Builders and UI Recorders

The ability to record and playback a user's interaction with a computer in a GUI environment has the potential to benefit multiple parties, including businesses (or other large organizations), users, software developers, testers, and computer support personnel. For example, business organizations can streamline a business process, such as use of software for supply-chain management, by automating much of the process. Users benefit by creating macros or scripts that combine a series of inputted actions into a playback action triggered in a single step. Software developers can use the ability to record user actions to help generate test cases for software under development. Testers can use record and playback tools to build tests to perform automated regression testing. Computer support personnel can record user actions to discover the reason for computer crashes or hangs, or to help users understand how to use software.

A. Commercial Macro Builders

Many conventional macro builder programs generate scripts that show the internal commands and actions taken by the computer or application to perform a function. However, in many instances, users must independently develop scripts based on a set of

scripting commands and complex programming constructs. Thus, users have to understand programming logic and, to some extent, the underlying logic of the programs being controlled to create and use a macro. For example, AutoMate, a macro program, uses a drag-and-drop task builder to create a macro script by dragging and dropping specific steps into the order they should be executed, which means the AutoMate user has to understand how and in what order commands should be issued. Macro Scheduler is a macro creation program that allows a user to write a macro script using more than 200 script commands and programming constructs (not including actual declared variables and other user-defined structures). The complexity required to create and edit the scripts generated by these macro programs and the fact that the scripts generated by these macro builders do not represent an actual step-by-step readable recording of UI activity lessen those macro programs' usefulness, particularly to novice users and to computer support personnel and software developers attempting to troubleshoot problems.

B. Internal Macro Languages

As an alternative to commercial macro building software, many applications have the built-in ability to record and playback macros using their own special-purpose application control language ("macro language"). An application's macro language is typically unique to the application and is generally based on the application's internal object model ("IOM"). While it is possible to build a macro language without object-oriented techniques, most internal models use an object-oriented representation of the structure of the program. The IOM provides an accessible outline or model of the classes, attributes, operations, parameters, relationships, and associations of the underlying objects for the program. Macro languages access their application's IOM and hook into its communications mechanisms (such as event calls) to access and call features within the application. For example, a user of Microsoft Word can record a macro to automatically format text. A main drawback of most macro languages, however, is that they are application specific. A macro recorded by one application generally is not supported by other applications, particularly if the two applications were developed by competing software companies. In some cases, a macro recorded for one version of an application is not supported by later versions of the application.

C. Dependence on Macros

Developing macros can be difficult, and many users and businesses are reluctant to do anything that might break their existing macros. Businesses in particular are reluctant to do anything that might negatively affect their business processes. In fact, many users and businesses are reluctant to upgrade software or even install patches for fear of "breaking" something. For example, suppose a business employs an automated business process (e.g., in the form of a macro) that scans received faxes, automatically performs optical character recognition ("OCR") on the fax to produce a text file version of the fax, compares the data in the text file to entries in a spreadsheet to verify account information, and finally sends a confirmation email to the sender of the fax. This business process most likely uses a combined set of potentially complicated macros and a variety of software packages to function properly (e.g., a scanner program, an OCR program, a spreadsheet program, a text file program, etc.). Businesses and users are often apprehensive about upgrading macros or software unless they have assurances that current investments into their automated processes will remain intact.

D. Other UI Recorders

Many conventional UI recorders have similar drawbacks to those of the macro tools described above in that they use complex scripting commands and programming constructs to represent data. Another drawback is that conventional playback is very dependent on the recording computer's pre-existing conditions. For example, playback may depend on a certain hardware configuration, software installation, and/or the dynamic state of the runtime environment (such as the availability or location of a UI element for a particular recorded interaction). Using a conventional playback tool, any changes to those pre-existing conditions can cause playback to fail.

As an example, suppose a user reconfigures an application user interface. Some GUI-based applications allow a user to move buttons, reconfigure menus, add or remove other UI elements for program features, add links to macros on a menu bar, or perform other UI alterations. Although the ability to personalize menus is a useful feature (e.g.,

enabling users to customize a UI to their specific needs), it may cause many recording and playback tools to fail because UI elements are not in their expected locations.

For example, the Microsoft Windows 3.1 Macro Recorder attempted to replicate user actions in a GUI by recording mouse movements and mouse coordinates when a mouse button was clicked. On playback, if a UI element was not where it was expected to be, playback failed. For additional information about macro recording in Microsoft Windows 3.1, see, for example, the reference entitled, User's Guide for Microsoft Windows for Workgroups, at page 137. As another example, suppose a user records UI activity at a low monitor/desktop resolution. Later, the user changes to a higher monitor/desktop resolution. In this case, as above, playback would most likely fail because the screen position of the UI elements has changed. Hence, as the computer environment changes playback becomes increasingly unreliable.

In conclusion, there is a need for simple, system-wide macro and UI recording tools that are compatible with existing macro languages and application-specific macro recorders. At the same time, there is a need for simple, system-wide macro and UI recording tools that work with dynamic user interfaces.

Summary

Techniques and tools are presented herein for recording and reliably replicating graphical user interface ("GUI") activity for various applications. For example, a macro and UI recording and playback tool simplifies the tasks of building macros and recording GUI activity for some applications while still supporting application-specific macro recorders for other applications, and also supporting macro and UI recording against dynamic user interfaces.

According to a first aspect, a tool receives internal macro actions from graphical user interface-based applications as opaque tokens. For example, a software program records user activity in its own native recording environment. Based on the recorded input, the software program generates opaque tokens that are transmitted to the tool.

According to a second aspect, a tool combines recorded tokens from internal macro actions with other user interface activity. For example, a software program passes recorded internal macro actions as opaque tokens to a tool. The tool combines those

20

opaque tokens with tokens received from a UI system recorder. Based on the combined tokens, the tool replicates user interface activity.

According to a third aspect, a tool collects recorded tokens associated with user interface activity from diverse applications and replicates the user interface activity by supplying each recorded token to the application with which each token is associated. For example, user interface activity spanning multiple applications is recorded as recorded tokens and combined by the tool into playback code. The tool replicates the recorded user activity by passing the recorded tokens from the playback code to the proper software programs.

10

According to a fourth aspect, a tool passes recorder tokens to a software program for playback by the internal recorder of the software program. For example, internal macro instructions are extracted from a set of recorder tokens and supplied to the internal recorder of the software program associated with the tokens for playback.

Additional features and advantages of the invention will be made apparent from the following detailed description of implementations that proceeds with reference to the accompanying drawing.

Brief Description Of The Drawings

Figure 1 is a block diagram of a user interface recording and playback framework.

20

Figure 2 is a block diagram of a system-wide UI recording tool.

25

Figure 3 is a flowchart illustrating a technique for recording application-native macro activity with a system-wide UI recording tool.

Figure 4 is a block diagram of a system-wide UI playback tool.

Figure 5 is a flowchart illustrating a technique for managing playback of application-native macro activity with a system-wide UI playback tool.

Figure 6 is a block diagram of a suitable computing environment for implementing system-wide UI recording and/or playback.

Detailed Description

The following description is directed to techniques and tools for macro and UI recording and playback for diverse GUI-based software. For example, a system-wide

21

21

macro and UI recording tool uses an appropriate recording technique depending on the type of user activity being recorded and the recording capabilities of the application being used (at times referred to herein as "smart" recording). The tool may use a UIA-based recording mechanism to record user activity against a UIA-compatible application. If the application is not UIA-compatible (or even if it is but recording with another mechanism is desired), the tool may record the raw steps (e.g., mouse button clicks, keystrokes) that a user performs interacting with the application. The tool may use a special recording mechanism in various circumstances. These circumstances include recording of activities described in an application-specific internal macro language (recorded by the application using its own internal recording mechanism) and recording of activities against a dynamic user interface. The tools described herein may record activity against various kinds of applications, operating system software, or other software.

U.S. Patent Application Serial No. aa/bbb,ccc, entitled "Recording/Playback Tools for UI-based Applications," filed June 15, 2004, the disclosure of which is hereby incorporated by reference, describes various recording and playback tools. Some of these tools use the UIA framework to record input based on a UI element's identification value ("UID"). UIA generates a composite ID that uniquely identifies a UI element in a GUI. The UID depends at least in part on location of the UI element in the user interface and its relationship with other UI elements. When a tool uses a UID to record and playback activities involving a specific UI element, changes to the element's location can cause playback to fail.

For example, suppose a user reconfigures a user interface by removing menu bar buttons. When a playback tool attempts to actuate the removed button, playback fails. As another example, often when an application is upgraded features are moved and/or renamed to accommodate new features. For instance, suppose in one version an "Open File" feature is located as the third option on a "File" menu. In a subsequent release, the "Open File" feature is moved to accommodate a "New File" feature, and "Open File" becomes the fourth item listed on the "File" menu. Macros based on the old user interface may break because the UI element moved and/or the UID has changed.

In some implementations, a system-wide macro and UI recording tool uses smart recording to address these kinds of problems. An application records user actions in the

application's internal macro language. After recording, the recorded steps expressed in the application's internal macro language are passed as opaque recorded step tokens from the application to the macro and UI recording tool. The system-wide macro and UI recording tool may then further encapsulate the recorded step tokens and/or associate the recorded step tokens with marking information. In some implementations, each macro instruction is converted into a separate recorded step token. Alternatively, a series of related macro language instructions are wrapped into one recorded step token. Those tokens maintain the temporal sequencing of the underlying macro language instructions so as to permit them to be played back by the application as they were recorded, while also allowing for interoperation with the generalized, system-wide macro and UI recording tool. After receiving the recorded step token, the tool optionally generates code that reflects the recorded steps. Subsequently, the code may be played back to recreate the recorded steps or, alternatively, the code may be edited for testing, troubleshooting, or other purposes.

For example, suppose a word processor has a macro language instruction "Open(filename)" that hooks into the application's internal object model to open a file with the name "file name." During recording with the smart recording tool, the macro instruction is passed as a recorded step token, associated with other appropriate information, and stored for playback, at which time the underlying instruction, "Open(filename)," is passed as a recorded step token from the smart recording tool back to the application. This permits applications to use their own native recording environments to record and playback UI activity within a system-wide UI recording and playback framework. A system-wide macro may be written that is not subject to the problems associated with menu customizations and changes in user interfaces associated with software upgrades, so long as the underlying macro language remains compatible with prior versions of the software. Moreover, the system-wide recording framework provides a unified recording and playback interface. It also simplifies the interleaving of macros and other UI recordings for different applications in combined macro and UI recording processes.

As part of the system-wide UI recording framework, when an application does not use an internal macro recorder or other application-specific UI recorder, a different

recording mechanism may be used. For example, a UIA-based recording system is used, as described in the U.S. patent application entitled "Recording/Playback Tools for UI-based Applications," which works well for recording user interface activity against applications having a static set of menus. Alternatively, if an application does not support operation according the UI Automation architecture, the system-wide UI recording framework RPF may simply record the raw steps a user performs as he or she interacts with the UI-based application. In some cases, the various recording techniques are used in combination. For example in some test cases the raw user input, the UIA-based data, and internal macro recorder data are all recorded.

Recording the raw steps (e.g., mouse clicks, keystrokes) that a user performs interacting with a UI-based application is relatively simple; the challenges are in how to represent and process this information when an application has a configurable user interface (e.g., one which allows the menus and menu items to be rearranged at will), when an application is upgraded modifying the user interface, or when an application processes macro or UI recording information in an internal language. Conceptually, some techniques and tools described herein improve recording performance by integrating the internal recording capabilities of the application being recorded with an abstract, user-friendly interface through which a user may convert recorded steps and input into playback code. As noted above, applications with their own internal recording capabilities may invoke their internal recorder to map a user's actions to the application's internal object model, thus solving the "upgrade lock-out" and menu reconfiguration problems caused by simple menu/menu-item recording.

In some implementations, application-specific macro or UI recording data is collected and then converted into recorded step tokens using tokenizing techniques. The techniques and tools create opaque tokens defining user interaction with an application. Once the recorded step tokens have been generated, other recorded actions (e.g., UIA events also in tokens) may be converted into programming code for playback on the same or another computer, potentially with application-generated recorded step tokens still embedded within the programming code.

In various common situations, the techniques and tools described herein allow a system-wide macro or UI recording tool to interoperate with various different kinds of

24

software programs (e.g., UIA compatible applications, applications with internal macro recorders, applications that are non-UIA-compliant) and various kinds of user interfaces (e.g., static, dynamic). These techniques and tools make recording reliable and robust by using an application's own internal recording capabilities when appropriate. Moreover, the tools and techniques provide users with a consistent macro or UI recording interface. The techniques and tools can be implemented in various ways, and may be used in combination or separately.

I. UI Recording and Playback Framework

10 A UI Recording and Playback Framework ("UI RPF") is an extensible framework designed to facilitate the recording and playback of UI activity against applications or other software programs. Pluggable modules into the framework include filters, aggregators, tokenizers, and code generators. Figure 1 shows an exemplary UI RPF 100 in which various techniques described herein may be implemented. Basically, the UI RPF 100 records (or works with other software that records) what a user is doing and reproduces (or works with other software that reproduces) his or her actions on the same or another computer. UI RPF 100 is a cross-application (e.g., system level) framework that can be used with multiple applications and in many configurations, and it is designed to facilitate recording and playback of various different kinds of user activity. It is a foundation upon which recording and playback tools can be built. The UI RPF 100 may use services provided by UI system architecture software 120, such as Microsoft UI Automation, to facilitate UI recording, but this is not a requirement of the framework 100.

Figure 1 shows various modules associated with the UI RPF 100. The UI RPF 100 interacts with the UI system architecture software 120, which converts recorded UI input data such as mouse clicks, keystrokes, and input from applications 125, 126 into data that is more meaningful for UI recording and playback. The framework 100 receives recorded steps from applications such as a text editor 125 and calculator 126 through the UI system architecture software 120, possibly passed as tokens encapsulating system user interface actions. In some implementations, the UI RPF 100 employs a UI recording and playback mechanism described in the U.S. patent application entitled,

25

“Recording/Playback Tools for UI-based Applications,” to record and playback UI activity received through the UI system architecture software 120. Alternatively, the UI RPF 100 uses a different UI recording and playback mechanism.

The modules of the UI RPF 100 also include a smart recording and playback module 110 for interoperating with applications that have their own native macro recorder. Bypassing the UI system architecture software 120, the framework 100 receives recorded steps or other macro information in recorded step tokens from applications such as word processor 115, spreadsheet 116, and photo editor 117. Applications 115, 116, 117 have their own internal recording capabilities to map a user's input to actions in the application's native recording environment, then the applications 115, 116, 117 pass the internally recorded actions as tokens to the framework 100 through the smart UI recording and playback module 110.

The UI RPF 100 maintains a list of the applications for which macro or UI activity may be recorded. A new application seen by the UI RPF 100 is queried, at least the first time the application is launched, to see if it supports interaction across a smart recording interface. If the application does support interaction across the smart interface, then user actions may be recorded according to the application's recording capabilities and passed back and forth through the interface. Alternatively, user actions are recorded through both the application's internal recording mechanism (with data passing through the smart recording interface) and through the UI system architecture software 120, where both options are possible. Or, even if both options are possible, a user, application, or system-wide tool may decide to use either the application's internal recording mechanism or recording through the UI system architecture software 120, but not both. Or, the user, application, or system-wide tool may decide to record the raw steps (e.g., mouse movements, mouse button clicks, keystrokes) associated with the user actions, either separately or in combination with the other recorded data.

When user activity recorded through an application's internal recording mechanism is in turn recorded by the UI RPF through the smart interface, the application converts the recorded user activity events into tokens. The contents of these tokens are defined in Visual Basic for Applications (“VBA”) or in some other language or object model used by the application. Ultimately, the information stored in the “payloads” of

these tokens is not interpreted by the UI RPF 100, but only by the respective applications for which the information is in a native format. The UI RPF 100 may further tokenize recorded step tokens and associate the data with other relevant data such as the name of the application.

5 In the system-wide tool, recorded step token data may be concatenated with recorded step token data obtained from other applications with native recorders, or with recorded data obtained through the UI system architecture software 120, so as to implement system level recording and playback.

10 In addition to recording macro and UI data, the framework 100 may produce playback code that replicates the recorded activity and/or encapsulates data that allows other software to replicate the recorded activity.

For example, in a Microsoft Windows environment, a user begins recording UI activity, then clicks on the Windows "Start" button, and launches a word processor that works with the smart recording interface. In the context of the UI RPF, user actions
15 including the mouse click on the "Start" button are recorded. When the word processor is launched, the UI RPF checks to see if the word processor is compatible with the smart recording interface. If it is, the UI RPF invokes the proper tools within the word processor itself to record user actions performed against it. If the word processor is also enabled to work with the UI system architecture software, the tool may also record user
20 actions (UI Automation events) in conjunction with the UI system architecture software. Additionally, the tool may also record the raw steps (e.g., mouse button clicks, keystrokes) associated with user actions, which may be particularly relevant if the word processor does not support the smart recording interface and is not enabled to work with the UI system architecture software. Thus, the UI RPF may use alternative mechanisms
25 for recording and playback.

This framework 100 may include elements of software and/or hardware. The relationships shown between modules in Figure 1 indicate the main flow of information; other relationships are not shown for the sake of simplicity. Depending on implementation, modules can be added, omitted, split into multiple modules, combined
30 with other modules, and/or replaced with like modules. Alternatively, a framework with

77

different modules and/or other configurations of modules performs one or more of the UI recording and playback techniques described herein.

II. Smart Recording

5 Figure 2 shows a system-wide UI recording tool 220 in conjunction with which various techniques described herein may be implemented. The UI RPF of Figure 1 includes the recording tool 220. Alternatively, the recording tool 220 is used as part of another UI framework.

10 The recording tool 220 is system-wide in that it records UI system architecture information for UI recording with certain applications, but also interoperates with applications that record user interface activities in their own native macro recording environments. Such native recording applications exchange recorded macro or UI data with the tool 220 across the special native recording interface (shown as the smart RPF API in Figure 2). This allows the tool 220 to use (and interoperate with) the application's
15 own recording capabilities, which may thus allow UI recording for dynamic user interfaces and/or UI recording that is re-playable for changed user interfaces for different versions of an application.

20 The recording tool 220 controls the mechanism(s) to be used when recording macro or UI data. For example, depending on user input, user settings, or other criteria, the tool 220 decides whether to use native application recording, UI system recording, or both for a particular application. The tool may also record the raw UI input from the user's mouse and keyboard, either separately or in combination with the other recording mechanisms. The tool 220 may concurrently record macro or UI data for each of multiple applications in use, or for selected applications among multiple applications in
25 use. For example, if a user configures the recording tool 220 to only record data from a word processor program, as a user records the cutting and pasting of data from a spreadsheet program to the word processor, only the steps related to the word processor are recorded.

78

70

A. Native Recording Path

For native macro or UI recording, a user interacts with the application 240 using user input devices such as a mouse, a keyboard, a touch pad, touch screen, Braille screen, joystick, microphone, or other similar device. As the user interacts with the application 240, the user inputs are processed and recorded by the macro/UI recorder 245 as steps in the application's native macro recording environment, as expressed in a language such as VBA, PerfectScript, M4, TeX, or other macro language. The application's internal recording capabilities provide a robust framework for acquiring data that may be predictably and reliably played back by the application. Recorded steps are only limited by the limitations of an application's native recording environment. For example, if an application permits a user to declare variables in a macro, those variables may be recorded and tokenized.

The application 240 tokenizes the recorded steps (shown as the tokenizer 247) so they can be passed to the tool 220. Then, the application passes the recorded step tokens to the tool 220 across the interface. The tokens are opaque to the tool 220, which means the tool 220 handles the recorded step tokens as discrete, opaque chunks of data whose contents are not interpreted by the tool 220. The application 240, on the other hand, may interpret and react to the contents of the recorded step tokens. The contents are in a native format for the application 240. For example, the application 240 encapsulates as a token recorded macro instructions to open a file exactly as they were recorded (e.g., as binary data, a text command, or a DLL instruction).

The application 240 converts each discrete recorded step into a token, or the application 240 groups multiple recorded steps as a single token. As for the mechanics of the tokenization, the application 240 may group recorded step data into one or more fields of a data structure for a token, which is then passed by copy or reference to the tool 220. Or, the application 240 may pass recorded step data as parameters of a method call, which the tool 220 receives and handles as opaque data. The application 240 may use an XML rules sheet (which defines a set of rules for tokenizing recorded step data) to reformat or reorganize the recorded step data into tokens. Or, the application 240 uses some other mechanism for tokenization. Conceptually, the tokenization may be viewed as placing the recorded step data in a token data structure, or wrapping the recorded step

X

data with token information. This “hides” the macro language instructions from the tool, creating a token that is macro language independent but whose contents are recognizable in the appropriate native recording environment.

5 The application 240 may prepend, append, or otherwise add marking data to the recorded step when tokenizing the data. The marking data provides relevant details for playback, for example, application information (e.g., the name of the application and its version), a timestamp to indicate an order, privileges information (e.g., what rights are needed to play the step), etc.

10 The tool 220 receives tokens (shown as the token acceptor 235) passed from one or more applications over the special native recording interface. While the tool 220 does not interpret the contents of the recorded step tokens, it may (instead of or in addition to the application) add additional layers of data to the recorded step tokens and/or associate the tokens with marking information such as application name, timestamps, index values, etc. so that the steps may be played back by the correct application and in the proper
15 sequence. In at least this sense, the tool 220 may be considered to also perform tokenization.

The application 240 supports the native recording interface in that it formats information for passing to the tool 220 across the interface, makes the appropriate calls across the interface, implements callback methods for the tool 220 to call, etc. The tool
20 220 supports the interface in the sense that it implements methods for applications to call, handles information passed in by applications as expected, calls callback methods of applications, etc. The exact call and processing mechanisms used for the interface are implementation dependent.

25 B. UI System Recording Path

The application 240 with the native macro recorder and/or another application 260 may support interaction with the system recording tool 220 through the UI system architecture software 120. For UI system architecture recording, various components 225
30 of the recording tool 220 collect (a) user input data, (b) monitor other UI activity (such as focus change, selection change, window creation, etc.), (c) track information about the environment such as running applications, OS version, localization information, default

UI language, current UI language, hardware devices and configurations, security modes, and/or other data that may help describe or recreate the proper environment for playback, (d) resolve the user input data to user interface elements affected by the user input, and (e) express the user activities as logical recording events. These logical recording events help provide readable UI recording output and reliable UI recording playback. Logical recording events may be considered as a type of UI recording token, but unlike the opaque recorded step tokens with native recording data, the tool 220 interprets the logical recording event tokens for viewing, editing, conversion to code, etc.

In some implementations, the tool 220 performs UI system architecture recording with filters and aggregators as described in the U.S. patent application entitled "Recording/Playback Tools for UI-based Applications." Alternatively, the tool 220 uses another technique for UI system architecture recording.

The tool 220 may display recorded UI system architecture steps in a viewer such as a debugger window, source code browser, or text editor so users may read and browse the data for errors. Typically, the tool 220 does not display information about the recorded step tokens with native recording data, as the contents of such tokens are opaque to the tool 220. When the recorded step tokens are associated with marking information such as application name, timestamps, action labels or other annotations from the application or user, the tool 220 may display such information. The data may similarly be saved to a file such as a log file for later analysis.

C. Other User Input

Various components 225 of the recording tool 220 may also collect raw user input data and similarly monitor other UI activity, track information about the environment such as running applications, etc. Recording these events helps in situations where an application is not compatible with the smart recording interface or the UI system architecture. These recorded actions may also be displayed in a viewer so users may read and browse the data for errors. Typically, the data from the raw user actions is less readable and reliable for playback than the other types of recorded tokens as explained above in the background. However, when the other recording techniques are inapplicable, this provides a mechanism for recording non-UIA and non-native recorder

81

application data in the system-wide UI RPF. The data recorded from raw user actions may similarly be saved to a file such as a log file for later analysis.

D. Playback Code

5 In some implementations, the tool 220 combines tokens from different applications and recording modes in a single file by interleaving the data. For example, recorded step tokens for native recording actions for different native recorders are combined into one input stream, along with tokens for UI system architecture actions and any other user input. Typically, user input is received in a serial manner and sequencing
10 of tokens in this manner a stream maintains proper ordering. Timestamps, index values, and application information may mark tokens to help maintain proper ordering. Or, the tool 220 creates separate files for different applications and recording modes.

For UI system action tokens, the tool 220 uses a code generator 230 to generate playback code 250 from the tokens. This can be done automatically or when the user
15 selects a "Generate Code" option in the tool 220. The playback code 250 may be output in a variety of formats, including C#, XML, C++, JAVA, or any other programming, scripting, macro, or document-oriented language, or other commands or instructions for a playback interface or other mechanism to drive UI-based applications. For example, the code generator 230 maps tokens for UI system actions to corresponding instructions in
20 the playback code 250. For additional details on code generation in a UI system architecture environment, see the U.S. patent application entitled, "Recording/Playback Tools for UI-based Applications," filed June 15, 2004.

The tool 220 optionally puts the recorded step tokens for native recording actions into playback code 250 such as a script file that maintains the recorded step tokens in
25 correct temporal sequence. Or, the tool 220 uses a code generator 230 to generate playback code 250 that includes the recorded step tokens. Notably, the recorded step tokens that encapsulate native recording data are opaque to the tool 220, so such tokens are typically passed through to the playback code 250, and are not interpreted by the tool 220 for conversion to other expressions.

30 The playback code 250 may be viewed, saved, edited, modified, added to, deleted from, compiled, or otherwise used prior to playback, with the expected limitations on the

82

extent to which the tool may view, modify, etc. opaque recorded step tokens. Being able to edit the playback code 250 provides users with a convenient and powerful mechanism for creating macros, generating automatic test cases, automating steps in a program, etc. The generated code may be edited to add conditional expressions, to change variables, to add loops, and other programming instructions to the code. For example, on playback, suppose a piece of playback code alters a system file and requires that a user to be logged on with Administrator rights. Accordingly, a conditional expression is added to the playback code, which checks for the appropriate rights before running the piece of code. As another example, suppose software developers want to stress test a program by repeating the same series of instructions over and over again, so, a "for loop" is added to the playback code, which causes the playback code to be repeated a certain number of times.

The playback code 250 may then be used to manage playback of the recorded activity, as described below. The code 250 may be played back on the same computer or, alternatively, on a different computer(s).

E. Tool Architecture and Uses

In general, the various modules of the tool 220 are pluggable, which means they may be replaced with other modules with similar functionality for different applications. For example, the code generator 230 in the tool 220 is pluggable, so a developer can reuse some parts of the tool 220 but change the way code is generated by switching code generators. The tool 220 is thus extensible; its functionality may be altered for different applications. In some implementations, the tool 220 presents a menu of options for pluggable modules, thus allowing a user to control loading of appropriate modules for the functionality desired by the user.

For example, in a "heavyweight" recording mode, the tool 220 collects detailed information about UI activities and comprehensively logs the information. This provides more reliable playback but is more likely to perturb execution of the software against which activity is recorded. Or, in a "lightweight" recording mode, the tool 220 collects less information and, therefore, is less likely to perturb execution of the software against which activity is recorded. Yet, this may still provide enough information for playback in

83

83

the same environment, for showing user activity to support personnel, or for other purposes.

Similarly, the tool 220 may record information in an interactive mode in which UI elements are highlighted by a bounding rectangle or flashing to show what is recorded.

- 5 Other flashing feedback may be used to show the user the applications for which UI data is being recorded. Or, in the interactive mode, the user is prompted to enter descriptive annotation information for each action or for actions that the tool 220 or application 240 are unable to resolve. Alternatively, the tool 220 may operate in the background to record UI activity without disturbing the user.

- 10 The tool 220 is responsible for loading other pluggable modules as needed (e.g., modules for collecting data, tokenizing data, generating code, etc.), enforcing security (e.g., checking for privileges), spawning recording threads, synchronizing activities between modules, building and maintaining queues used for recording, logging errors (e.g., to a local file or optionally to a network service or server), and collecting
- 15 information about the recordings (e.g., OS version, localization information, hardware configuration, etc.).

- The relationships shown between modules in Figure 2 indicate the main flow of information; other relationships are not shown for the sake of simplicity. Depending on implementation, modules can be added, omitted, split into multiple modules, combined
- 20 with other modules, and/or replaced with like modules. Alternatively, a tool with different modules and/or other configurations of modules performs one or more of the recording techniques described herein.

- The user interface of the tool 220 may be a VCR-like interface positioned around a recording window, docked to the top or bottom of a computer screen, or minimized to a
- 25 task bar. The interface includes buttons for features such as "Record," "Pause," "Stop," and "Play," as well as an area for presenting description of recorded steps when such description is available. For examples of such interfaces, see the U.S. patent application entitled "Recording/Playback Tools for UI-based Applications."

F. Recording Technique, Timing of Recording

There are two modes for transferring the tokens to a tool 220 during recording: batch mode and interactive mode. In general, in batch mode, recorded step tokens are collected at the application and delivered to the tool 220 when requested. These requests
 5 to retrieve the collected tokens may occur when focus has changed away from the recording application, and may also occur periodically to distribute the recording load.

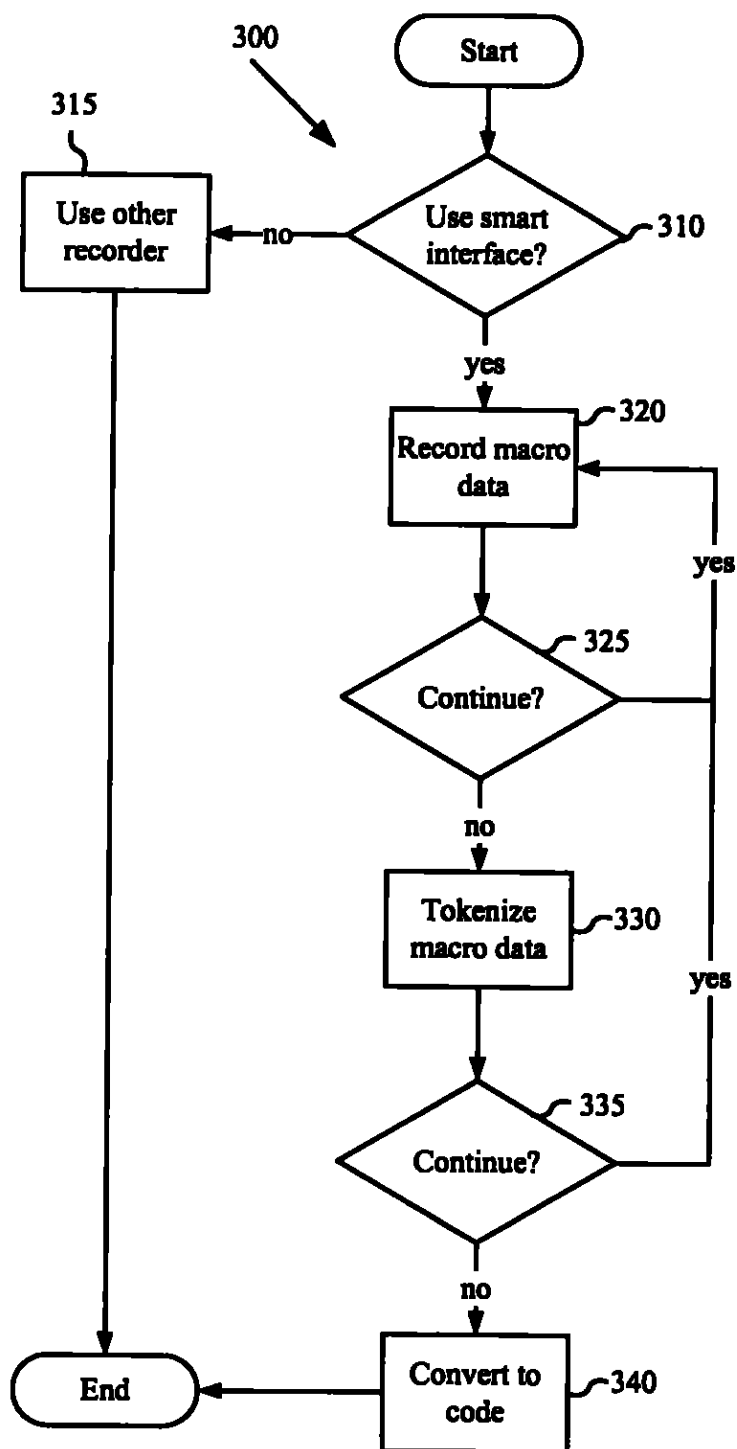
In interactive mode, in contrast, tokens are sent from the application being recorded to the tool 220 as they occur. One disadvantage of sending tokens interactively is that it increases the cross-process loading on the CPU. In a particular situation, either
 10 batch or interactive mode may be preferable to the other, depending on the application being recorded, the amount of data being recorded, and how quickly recorded step tokens are being generated.

Figure 3 shows a technique 300 for recording application-native macro activity with a system-wide UI recording tool. A tool and application such as the recording tool
 15 220 and application 240 shown in Figure 2 perform the technique 300. Alternatively, another tool and/or application perform the technique 300.

The tool checks 310 to see if the application uses a smart native recording interface. Some applications do not support interaction over the interface. Other applications might support interaction over the interface, but the user or tool determines
 20 not to use native recording. For example, when a user first launches an application the tool checks to see if the application supports interaction over the smart native recording interface, and if so, adds the name of the application to a list of applications that support interaction over the interface. Or, the tool checks to see if an application supports the smart native recording interface at the time recording begins. If an application does not
 25 support interaction over the interface, the tool may add the application to a list of non-supported applications

If the smart native recording interface is not used, the tool uses 315 another recorder to record UI activity. For example, the tool uses a UIA software recorder for a
 30 UIA-enabled application, or alternatively, a raw UI input recorder for non-UIA-enabled applications. While Figure 3 shows alternative paths of recording, the tool may instead use both the native recording and the other recording concurrently.

Figure 3



ef

Figure 4

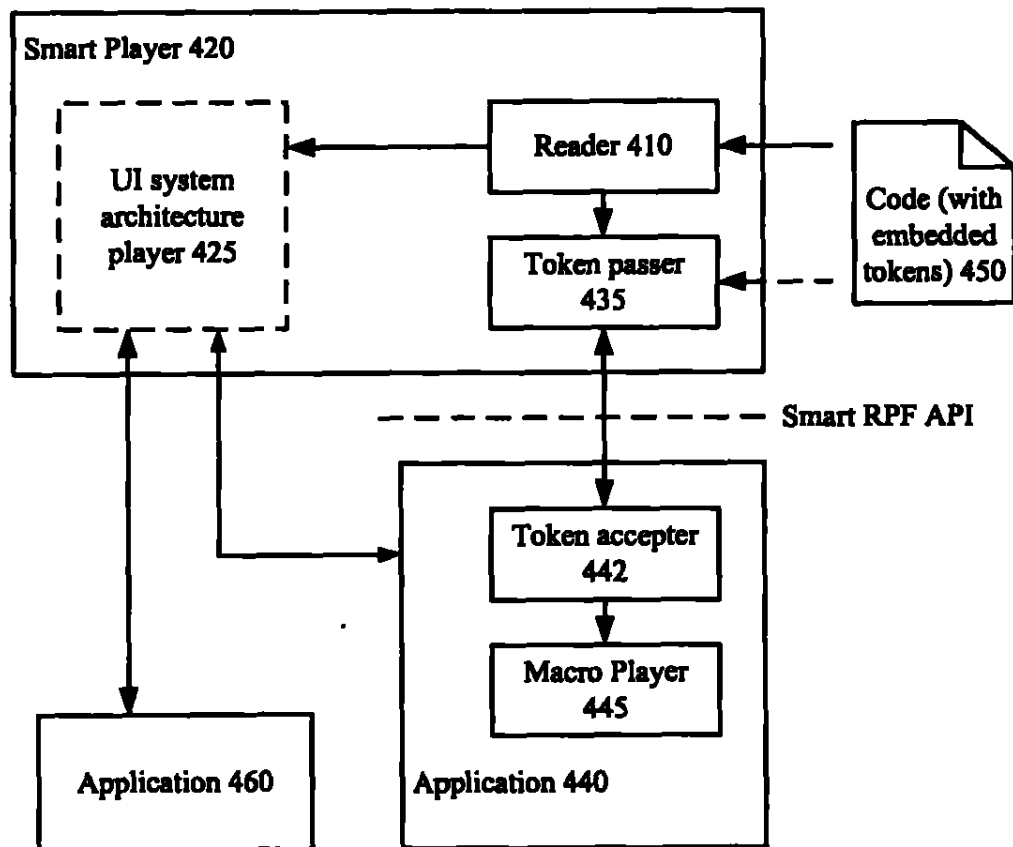


Figure 5

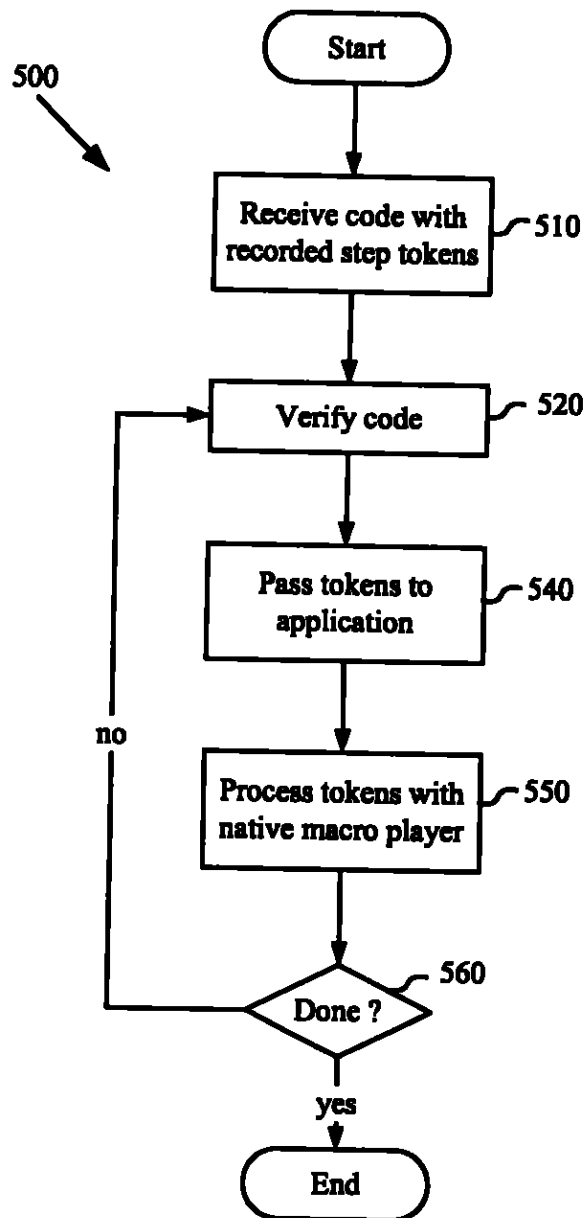


Figure 6

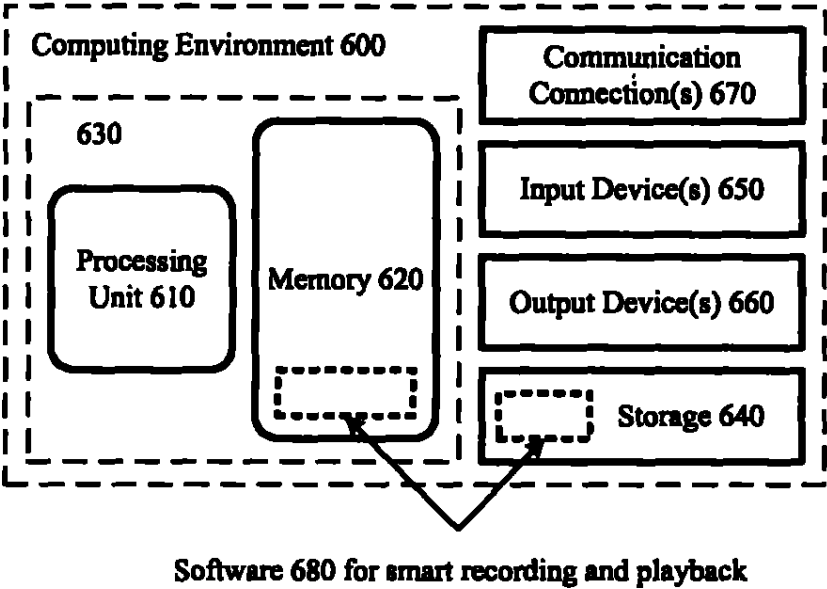


Figure 1

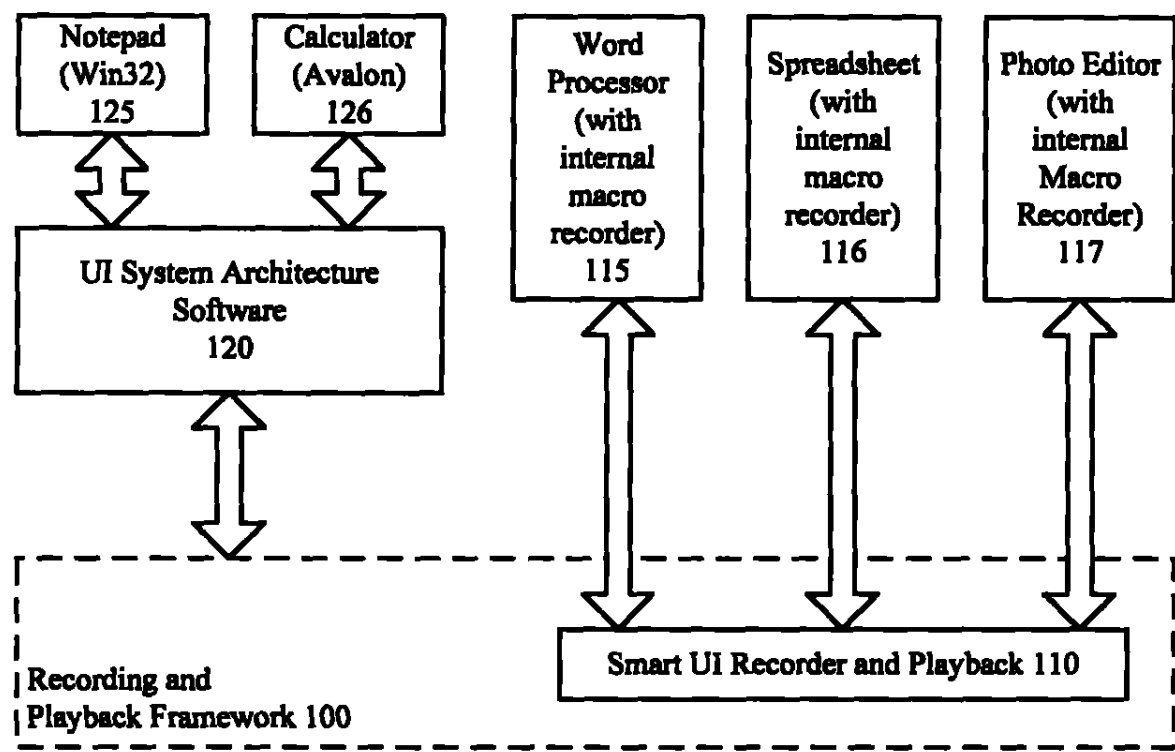


Figure 2

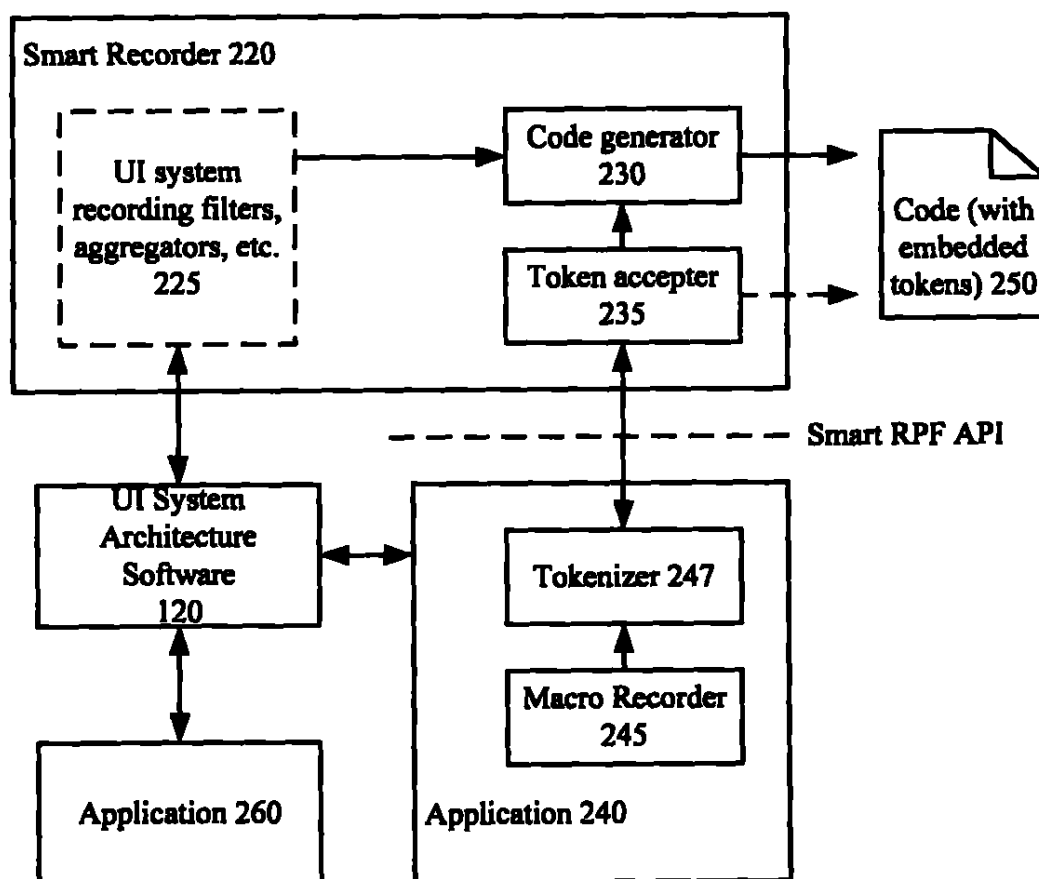
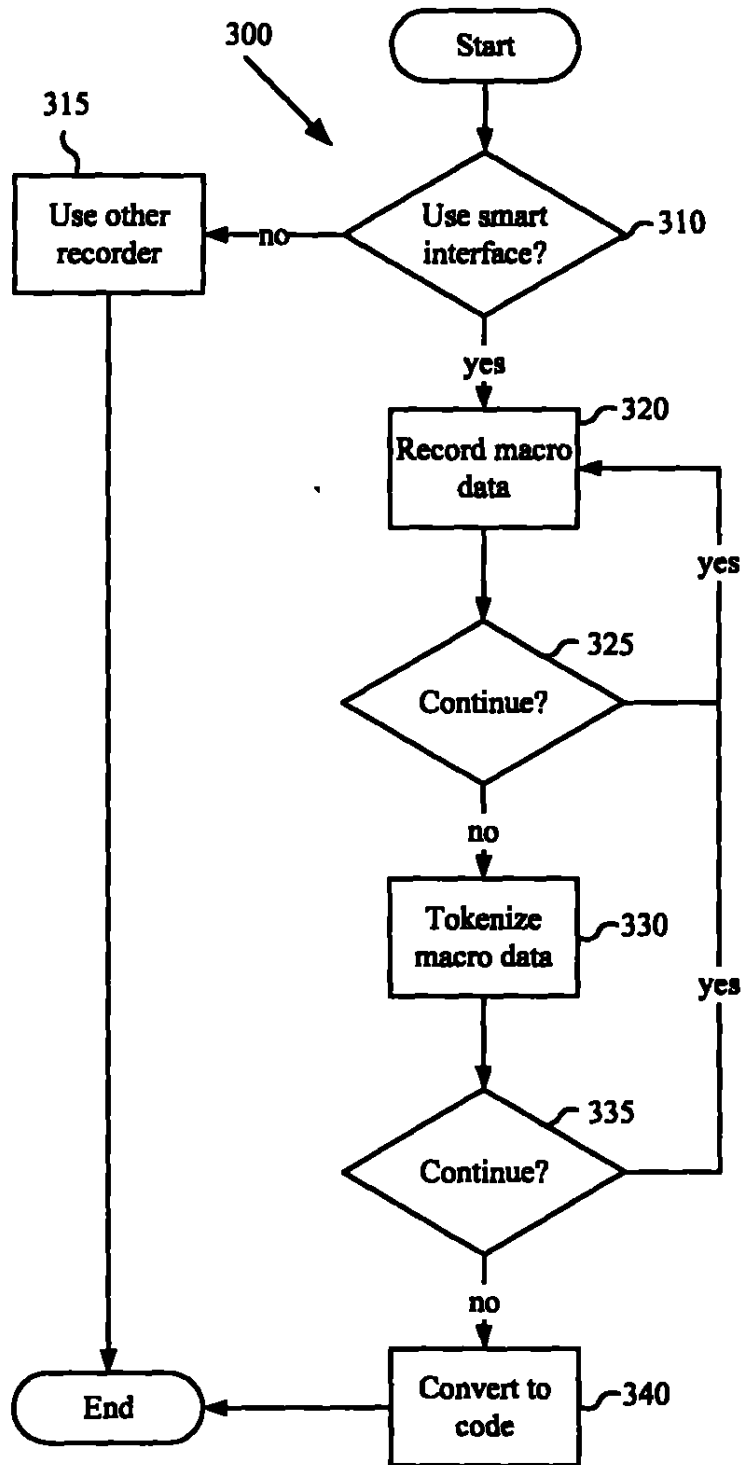


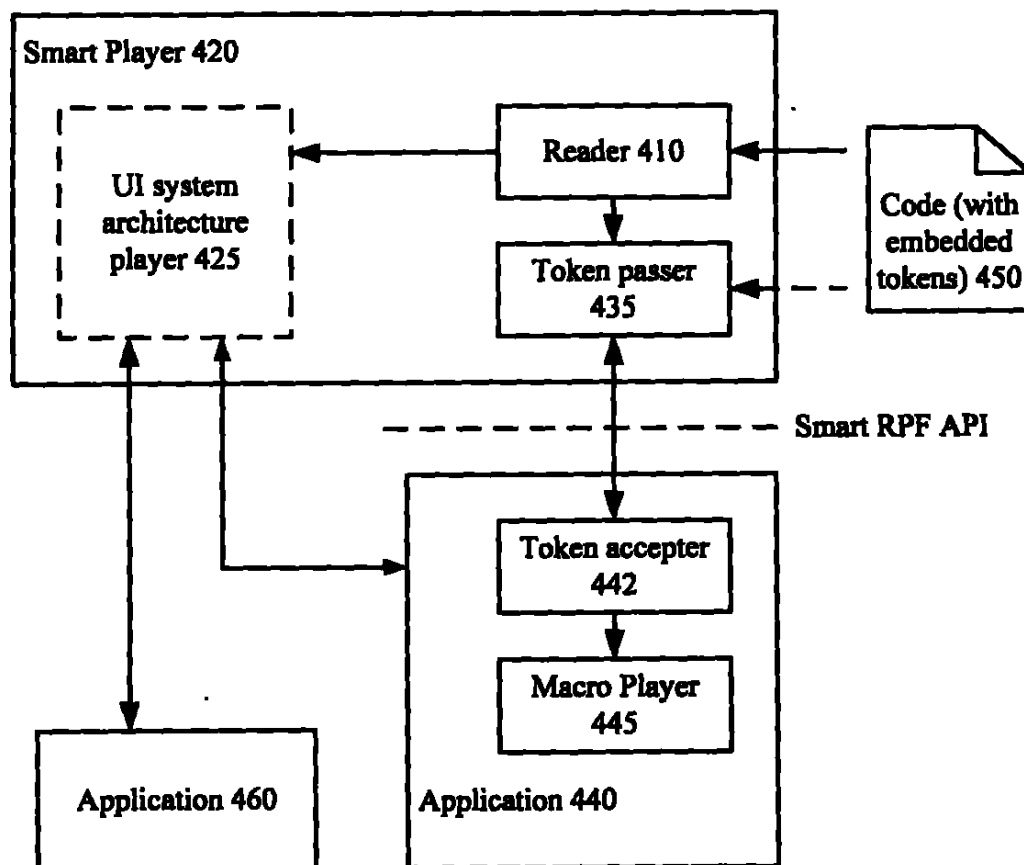
Figure 3



gr

92

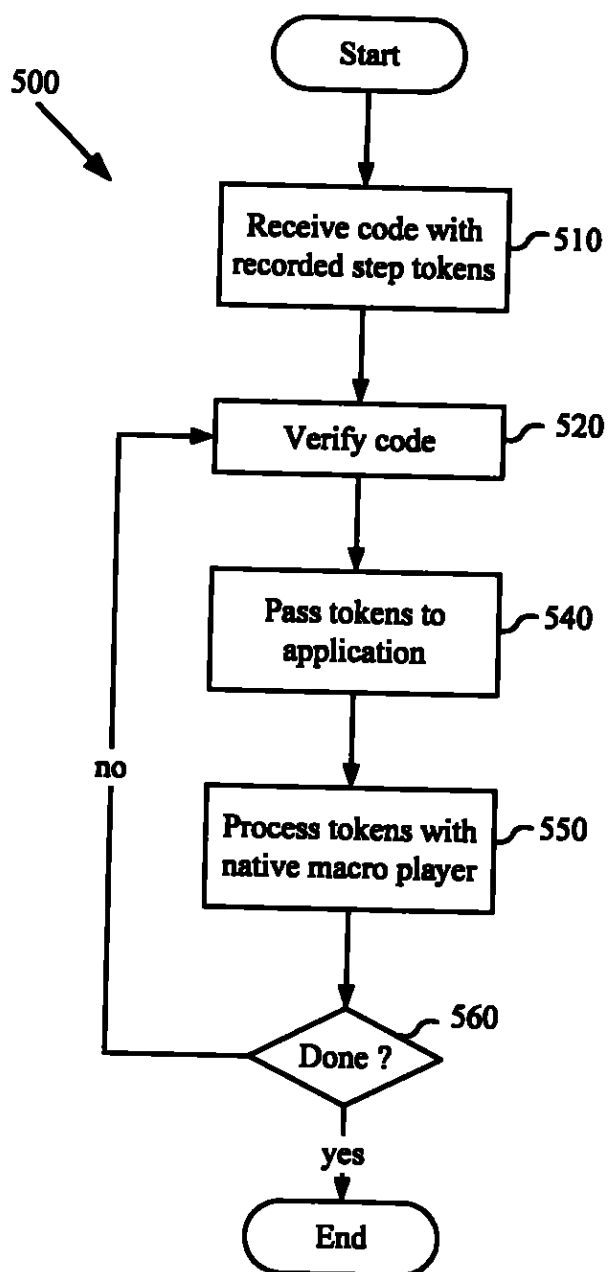
Figure 4



93

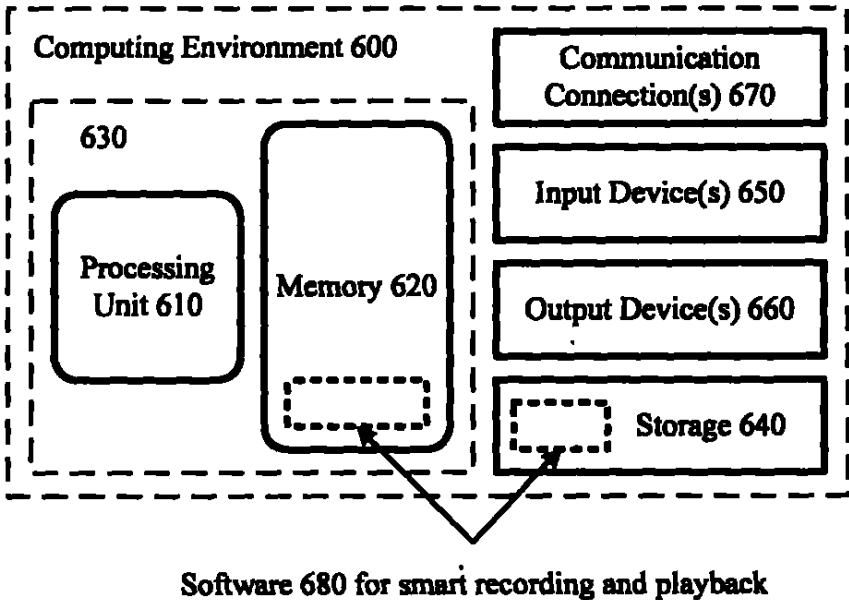
93

Figure 5



94

Figure 6



95

PA 1290472

THE UNITED STATES OF AMERICA

TO ALL TO WHOM THESE PRESENTS SHALL COME:

UNITED STATES DEPARTMENT OF COMMERCE

United States Patent and Trademark Office

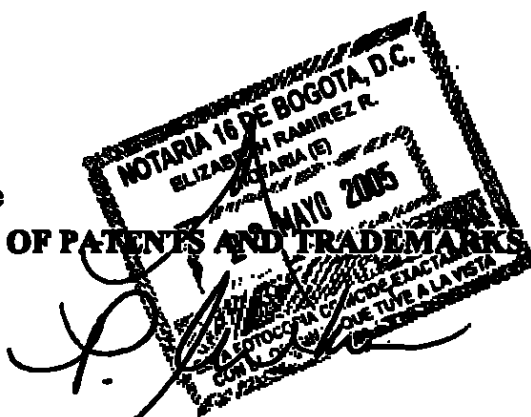
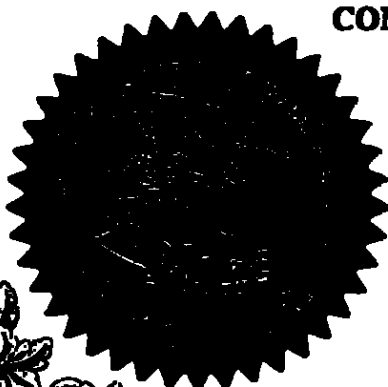
March 23, 2005

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: 10/382,861

FILING DATE: June 30, 2004

By Authority of the
COMMISSIONER OF PATENTS AND TRADEMARKS



P. SWAIN
Certifying Officer

95

22713 U.S. PTO

MO:cm 06/30/04 289761.doc
Attorney Reference Number 3382-67840-01
PATENT

EXPRESS MAIL LABEL NO. EV 331582587 US
DATE OF DEPOSIT: June 30, 2004

19587 U.S. PTO
10/882861

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

COMMISSIONER FOR PATENTS
P.O. BOX 1450
ALEXANDRIA, VA 22313-1450

Transmitted herewith for filing is the patent application of:

Inventor(s): Howard B. Herdeg, III

For: SMART UI RECORDING AND PLAYBACK FRAMEWORK

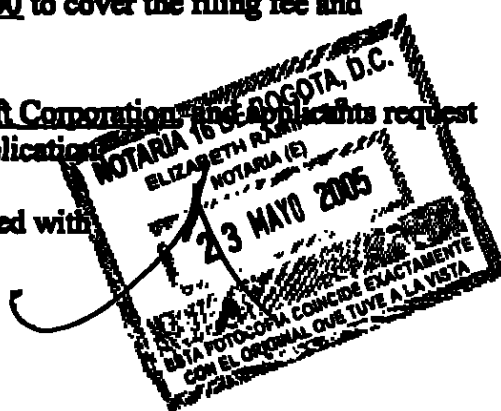
Enclosed are:

- ☒ Application Data Sheet.
- ☒ 33 pages of specification, 5 pages of claims, an abstract and a Combined Declaration and Power of Attorney.
- ☒ 6 sheet(s) of drawings.
- ☒ An assignment of the invention to: Microsoft Corporation, a Recordation Cover Sheet, and the Recordal fee of \$40.00.
- ☒ Information Disclosure Statement.
- ☒ Form PTO-1449 and copies of documents listed thereon.

FILING FEE					
For	Claims Filed	Number Free	Number Extra	Rate	Basic Fee
					\$770.00
Total Claims	27	20	- 7	\$18.00	\$126.00
Independent Claims	4	3	- 1	\$86.00	\$86.00
Multiple Dependent Claim Fee				\$290.00	
TOTAL FILING FEE					\$982.00

- ☒ PTO-2038 Credit Card Payment Form for \$1,022.00 to cover the filing fee and Assignment Recordal fee is enclosed.
- ☒ Assignee for this application is or will be Microsoft Corporation and applicants request that this information be listed on the published application.
- ☒ Address all correspondence to the address associated with

CUSTOMER NUMBER 26119



TRANSMITTAL - Page 1 of 2

9/5

JMO:lem 06/30/04 289761.doc
Attorney Reference Number 3382-67840-01
PATENT

97
EXPRESS MAIL LABEL NO. EV 331582587 US
DATE OF DEPOSIT: June 30, 2004

- ☒ The Director is hereby authorized to charge any additional fees that may be required in connection with the filing of this application and recording any assignment filed herewith, or credit over-payment, to Account No. 02-4550. A copy of this sheet is enclosed.
- ☒ Please return the enclosed postcard to confirm that the items listed above have been received.

Respectfully submitted,

KLARQUIST SPARKMAN, LLP

By


Kyle B. Rinehart
Registration No. 47,027

One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Telephone: (503) 226-7391
Facsimile: (503) 228-9446

cc: Docketing
Client (308158.01)



TRANSMITTAL - Page 2 of 2

Application Data Sheet

Application Information

Application number::

Filing Date::

Application Type::

Regular

Subject Matter::

Utility

Suggested classification::

Suggested Group Art Unit::

CD-ROM or CD-R?::

None

Number of CD disks::

Number of copies of CDs::

Title::

**SMART UI RECORDING AND PLAYBACK
FRAMEWORK**

Attorney Docket Number::

3382-67840-01

Request for Early Publication?::

NO

Request for Non-Publication?::

NO

Suggested Drawing Figure::

2

Total Drawing Sheets::

6

Small Entity?::

NO

Petition included?::

NO

Petition Type::

Licensed US Govt. Agency::

Contract or Grant Numbers::

Secrecy Order in Parent Appl.?::

NO

Applicant Information

Applicant Authority Type::

Inventor

Primary Citizenship Country::

US

Exp. Mail No. EV 331582587 US

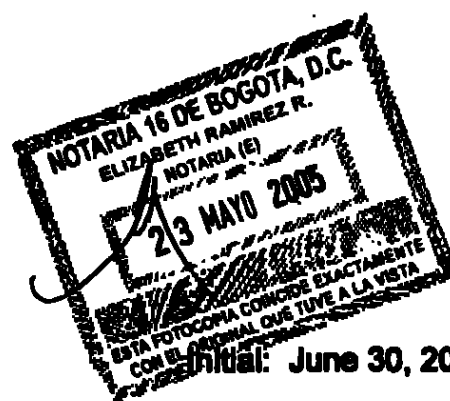
Initial: June 30, 2004



99

Status:: Full Capacity
Given Name:: Howard
Middle Name:: B.
Family Name:: Herdeg
Name Suffix:: III
City of Residence:: North Bend
State or Province of Residence:: WA
Country of Residence:: US
Street of mailing address:: 44415 166th Street SE
City of mailing address:: North Bend
State or Province of mailing address:: WA
Country of mailing address:: US
Postal or Zip Code of mailing address:: 98045

Exp. Mail No. EV 331582587 US



Initial: June 30, 2004

100

100

Correspondence Information

Correspondence Customer Number:: 26119

Representative Information

Representative Customer Number:: 26119

Domestic Priority Information

Application::	Continuity Type::	Parent Application::	Parent Filing Date::

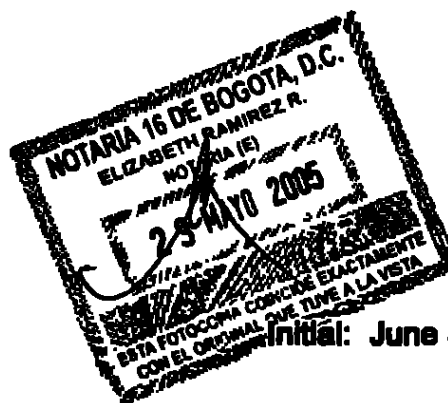
Foreign Priority Information

Country::	Application number::	Filing Date::	Priority Claimed::

Assignee Information

Assignee name:: Microsoft Corporation
Street of mailing address:: One Microsoft Way
City of mailing address:: Redmond
State or Province of mailing address:: WA
Country of mailing address:: USA
Postal or Zip Code of mailing address:: 98052

Exp. Mail No. EV 331582587 US



Initial: June 30, 2004

101

Smart UI Recording and Playback Framework

Technical Field

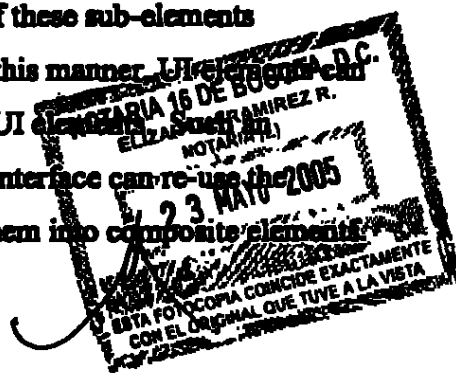
Tools and techniques are described for recording user activity within various graphical user interface based applications. For example, a system-wide recording tool supports application-native macro or UI recording as well as non-native macro or UI recording.

Background

10 I. Graphical User Interface

On many modern computer systems, users interact with software programs through a graphical user interface ("GUI"). Basically, a GUI is an interface between computer and user that uses pictures rather than just words to solicit user input and present the output of a program. The typical GUI is made up of user interface elements ("UI elements"), which are those aspects of a computer system or program that are seen, heard, or otherwise perceived or interacted with by a user. For example, UI elements include items such as icons, buttons, dialog boxes, edit boxes, list boxes, combo boxes, scroll bars, pick lists, pushbuttons, radio buttons and various components of World Wide Web pages (e.g., hyper-links and images). In a typical computer program it is common to encounter literally thousands of UI elements.

Although an individual element of a GUI may appear to the user as a single item, it may actually consist of a number of separate items or sub-elements that have been combined together. For example, a toolbar item may consist of a list element, a combo box element, a scroll bar element, etc. Furthermore, each of these sub-elements themselves may be composed from other sub-elements. In this manner, UI elements can serve as building blocks for building other, more complex, UI elements. This approach is useful because the software managing the user interface can re-use the definitions of certain common elements when assembling them into composite elements.

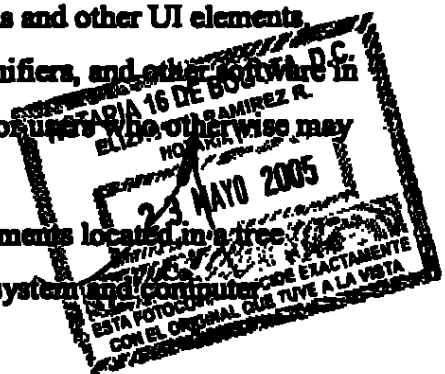


Many UI elements in a GUI environment represent features of a program and are displayed on the computer screen so users can interact with the program by selecting, highlighting, accessing, and operating them. This user interaction is done by maneuvering a pointer on the screen (typically controlled by a mouse or keyboard) and pressing or clicking buttons while the pointer is pointing to a UI element. For example, in a word processor, a user can maneuver the mouse to select an item on the program's menu bar, to click an icon from the tool bar, or to highlight blocks of text in the viewer window. Similarly, a user can use keyboard input to interact with a computer application. For instance, in the word processing program, a user can press "ALT-F," "CTRL-B," or other predefined keystroke combinations to access program features. Based on the input from the mouse or keyboard, the computer adds, changes, and manipulates what is displayed on the screen. GUI technologies provide convenient, user-friendly environments for users to interact with computer systems.

15 II. UI Automation

UI Automation ("UIA") is an accessibility framework for Microsoft Windows intended to address the needs of assistive technology products and automated test frameworks by providing programmatic access to information about the user interface ("UI"). For example, UIA allows a screen reader program to access information about the UI of a word processor, providing the reader program with the information it needs to provide audible cues to a visually impaired user. Through an application programming interface ("API") set of methods, UIA provides a well-structured mechanism for creating and interacting with a UI. Control and application developers use the UIA API set to make their products more accessible to different users through existing or new software (potentially written by other people) to access program menus and other UI elements. For example, braille screens, screen readers (narrators), magnifiers, and other software in Microsoft Windows can use UIA to facilitate computer use for users who otherwise may not have access.

In practice, UI Automation uses a hierarchy of UI elements located in a tree structure to provide reliable UI information to the operating system and computer.



103

applications. Elements of a GUI can be considered as nested within each other in order to accurately describe their organization. For example, at the very top of a tree structure is a desktop element that is representative of the GUI's "desktop" or default background area. The desktop element has within it several application elements for application

5 programs that have been invoked and that are ready to execute according to a user's instructions (e.g., a typical Microsoft Windows desktop may have several instances of applications such as Microsoft Word, Microsoft Excel, etc. loaded and ready to execute). At a lower level in the tree structure hierarchy are frames associated with an application (e.g., a word processor application may have several frames visible to a user at any given

10 time). Within each of the frames may be several documents, each document containing within it several UI elements (e.g., buttons, listboxes, etc.). UI elements may themselves be composites of other UI elements. For example, a dialog box or a combo box contains other UI elements such as a button control. Furthermore, the button element may contain yet other UI elements. Such nesting can be arbitrarily deeper and include an arbitrary

15 number of branches depending on the user interface and its component elements.

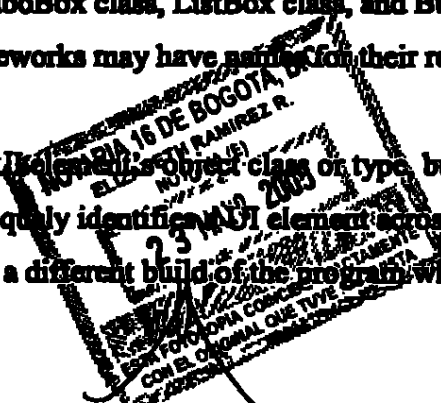
For some operating system platforms, an instance of a UI element is assigned an identifier to help distinguish that particular UI element from other UI elements. For example, in a Microsoft Windows based operating system, applications are associated with module identifiers that identify applications within a given desktop context. Also,

20 some user interface platforms (e.g., Microsoft Windows, Swing for Java) use a numeric identifier (control ID) for certain UI elements. In some computing environments, such as a Microsoft Windows environment, UI elements are often associated with a class name associated with the control class to which they belong. For instance, in a Microsoft Windows based system, common UI elements such as combo box, list box, and button

25 are associated with class names such as ComboBox class, ListBox class, and Button class, respectively. Similarly, other UI frameworks may have names for their respective classes of UI elements.

Notably, these techniques identify a UI element's object class or type, but do not singularly provide a strong identifier that uniquely identifies a UI element across a reboot

30 of the computer running the program, across a different build of the program when still in



104

104

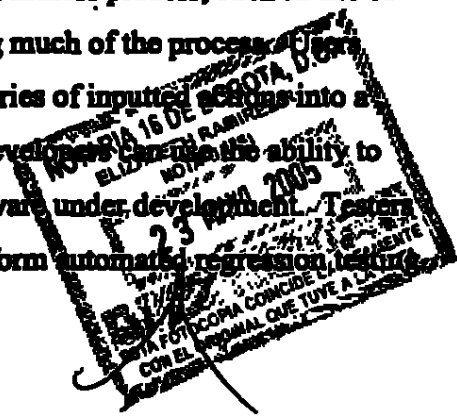
development, across the opening of an another instance of the same program, or for opening of the same program on another computer.

UIA overcomes these deficiencies by generating a composite ID that uniquely identifies a UI element in a GUI tree. UIA generates the composite identifier by adding
5 identification information (e.g., control name or control type) that is directly associated with a UI element to hierarchical identification information (e.g., parent control, child control, and/or sibling controls) and control pattern-specific information (e.g., depth of the UI element in the tree). For example, an identifier for a target UI element may be
10 generated by collecting identifying information related to parent UI elements that describe the hierarchical arrangement between a target leaf UI element and the root element at desktop. Through the concept of a path, the related identifiers for a UI element's unique hierarchy and parentage can be leveraged to identify it uniquely and persistently.

The unique identifier (persistent ID) provides easy access to individual UI
15 elements so that the functionality of a program hosting UI elements can be programmed and tested, and so that a particular UI element can be identified to other program modules. For additional information about UIA, see, for example, the documentation available through the Microsoft Developer Network.

20 III. Macros Builders and UI Recorders

The ability to record and playback a user's interaction with a computer in a GUI environment has the potential to benefit multiple parties, including businesses (or other large organizations), users, software developers, testers, and computer support personnel. For example, business organizations can streamline a business process, such as use of
25 software for supply-chain management, by automating much of the process. Users benefit by creating macros or scripts that combine a series of inputted actions into a playback action triggered in a single step. Software developers benefit from the ability to record user actions to help generate test cases for software under development. Testers can use record and playback tools to build tests to perform automated regression testing.



104

Computer support personnel can record user actions to discover the reason for computer crashes or hangs, or to help users understand how to use software.

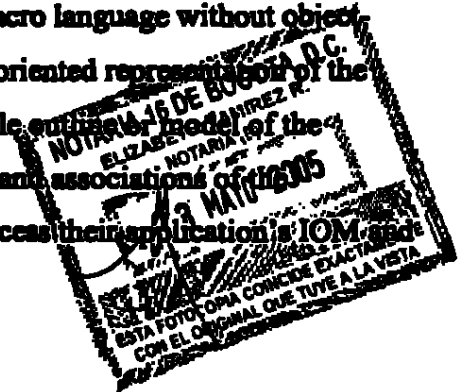
A. Commercial Macro Builders

Many conventional macro builder programs generate scripts that show the internal
5 commands and actions taken by the computer or application to perform a function.
However, in many instances, users must independently develop scripts based on a set of
scripting commands and complex programming constructs. Thus, users have to
understand programming logic and, to some extent, the underlying logic of the programs
being controlled to create and use a macro. For example, AutoMate, a macro program,
10 uses a drag-and-drop task builder to create a macro script by dragging and dropping
specific steps into the order they should be executed, which means the AutoMate user has
to understand how and in what order commands should be issued. Macro Scheduler is a
macro creation program that allows a user to write a macro script using more than 200
script commands and programming constructs (not including actual declared variables
15 and other user-defined structures). The complexity required to create and edit the scripts
generated by these macro programs and the fact that the scripts generated by these macro
builders do not represent an actual step-by-step readable recording of UI activity lessen
those macro programs' usefulness, particularly to novice users and to computer support
personnel and software developers attempting to troubleshoot problems.

20

B. Internal Macro Languages

As an alternative to commercial macro building software, many applications have
the built-in ability to record and playback macros using their own special-purpose
application control language ("macro language"). An application's macro language is
25 typically unique to the application and is generally based on the application's internal
object model ("IOM"). While it is possible to build a macro language without object-
oriented techniques, most internal models use an object-oriented representation of the
structure of the program. The IOM provides an accessible, graphical model of the
classes, attributes, operations, parameters, relationships, and associations of the
30 underlying objects for the program. Macro languages access their application's IOM and



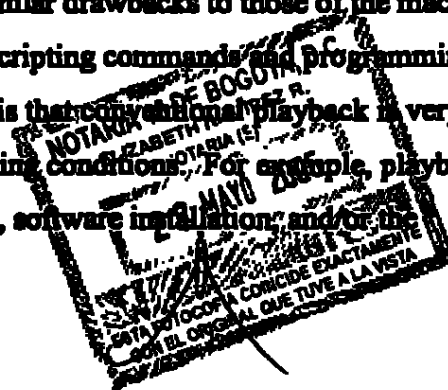
hook into its communications mechanisms (such as event calls) to access and call features within the application. For example, a user of Microsoft Word can record a macro to automatically format text. A main drawback of most macro languages, however, is that they are application specific. A macro recorded by one application generally is not supported by other applications, particularly if the two applications were developed by competing software companies. In some cases, a macro recorded for one version of an application is not supported by later versions of the application.

C. Dependence on Macros

Developing macros can be difficult, and many users and businesses are reluctant to do anything that might break their existing macros. Businesses in particular are reluctant to do anything that might negatively affect their business processes. In fact, many users and businesses are reluctant to upgrade software or even install patches for fear of "breaking" something. For example, suppose a business employs an automated business process (e.g., in the form of a macro) that scans received faxes, automatically performs optical character recognition ("OCR") on the fax to produce a text file version of the fax, compares the data in the text file to entries in a spreadsheet to verify account information, and finally sends a confirmation email to the sender of the fax. This business process most likely uses a combined set of potentially complicated macros and a variety of software packages to function properly (e.g., a scanner program, an OCR program, a spreadsheet program, a text file program, etc.). Businesses and users are often apprehensive about upgrading macros or software unless they have assurances that current investments into their automated processes will remain intact.

D. Other UI Recorders

Many conventional UI recorders have similar drawbacks to those of the macro tools described above in that they use complex scripting commands and programming constructs to represent data. Another drawback is that conventional playback is very dependent on the recording computer's pre-existing conditions. For example, playback may depend on a certain hardware configuration, software installation, and/or the



dynamic state of the runtime environment (such as the availability or location of a UI element for a particular recorded interaction). Using a conventional playback tool, any changes to those pre-existing conditions can cause playback to fail.

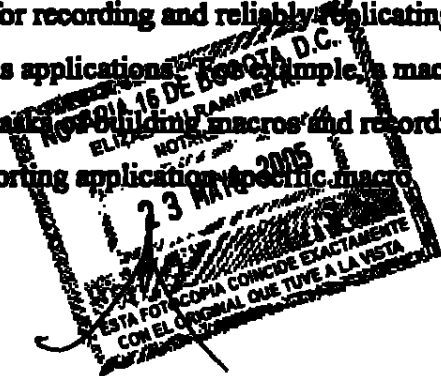
As an example, suppose a user reconfigures an application user interface. Some GUI-based applications allow a user to move buttons, reconfigure menus, add or remove other UI elements for program features, add links to macros on a menu bar, or perform other UI alterations. Although the ability to personalize menus is a useful feature (e.g., enabling users to customize a UI to their specific needs), it may cause many recording and playback tools to fail because UI elements are not in their expected locations.

For example, the Microsoft Windows 3.1 Macro Recorder attempted to replicate user actions in a GUI by recording mouse movements and mouse coordinates when a mouse button was clicked. On playback, if a UI element was not where it was expected to be, playback failed. For additional information about macro recording in Microsoft Windows 3.1, see, for example, the reference entitled, *User's Guide for Microsoft Windows for Workgroups*, at page 137. As another example, suppose a user records UI activity at a low monitor/desktop resolution. Later, the user changes to a higher monitor/desktop resolution. In this case, as above, playback would most likely fail because the screen position of the UI elements has changed. Hence, as the computer environment changes playback becomes increasingly unreliable.

In conclusion, there is a need for simple, system-wide macro and UI recording tools that are compatible with existing macro languages and application-specific macro recorders. At the same time, there is a need for simple, system-wide macro and UI recording tools that work with dynamic user interfaces.

Summary

Techniques and tools are presented herein for recording and reliably replicating graphical user interface ("GUI") activity for various applications. For example, a macro and UI recording and playback tool simplifies the task of building macros and recording GUI activity for some applications while still supporting application-specific macro



108

recorders for other applications, and also supporting macro and UI recording against dynamic user interfaces.

According to a first aspect, a tool receives internal macro actions from graphical user interface-based applications as opaque tokens. For example, a software program

5 records user activity in its own native recording environment. Based on the recorded input, the software program generates opaque tokens that are transmitted to the tool.

According to a second aspect, a tool combines recorded tokens from internal macro actions with other user interface activity. For example, a software program passes recorded internal macro actions as opaque tokens to a tool. The tool combines those

10 opaque tokens with tokens received from a UI system recorder. Based on the combined tokens, the tool replicates user interface activity.

According to a third aspect, a tool collects recorded tokens associated with user interface activity from diverse applications and replicates the user interface activity by supplying each recorded token to the application with which each token is associated. For

15 example, user interface activity spanning multiple applications is recorded as recorded tokens and combined by the tool into playback code. The tool replicates the recorded user activity by passing the recorded tokens from the playback code to the proper software programs.

According to a fourth aspect, a tool passes recorder tokens to a software program

20 for playback by the internal recorder of the software program. For example, internal macro instructions are extracted from a set of recorder tokens and supplied to the internal recorder of the software program associated with the tokens for playback.

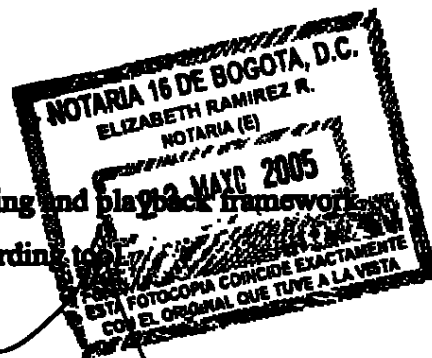
Additional features and advantages of the invention will be made apparent from the following detailed description of implementations that proceeds with reference to the

25 accompanying drawing.

Brief Description Of The Drawings

Figure 1 is a block diagram of a user interface recording and playback framework.

Figure 2 is a block diagram of a system-wide UI recording tool.



109

Figure 3 is a flowchart illustrating a technique for recording application-native macro activity with a system-wide UI recording tool.

Figure 4 is a block diagram of a system-wide UI playback tool.

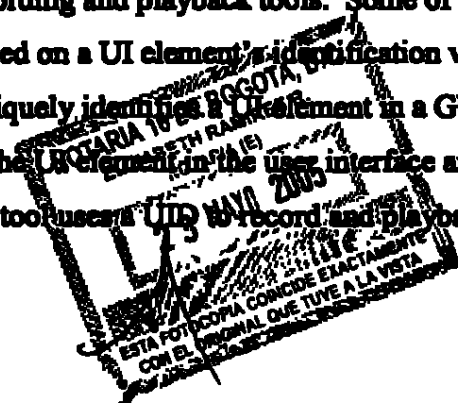
Figure 5 is a flowchart illustrating a technique for managing playback of application-native macro activity with a system-wide UI playback tool.

Figure 6 is a block diagram of a suitable computing environment for implementing system-wide UI recording and/or playback.

Detailed Description

The following description is directed to techniques and tools for macro and UI recording and playback for diverse GUI-based software. For example, a system-wide macro and UI recording tool uses an appropriate recording technique depending on the type of user activity being recorded and the recording capabilities of the application being used (at times referred to herein as "smart" recording). The tool may use a UIA-based recording mechanism to record user activity against a UIA-compatible application. If the application is not UIA-compatible (or even if it is but recording with another mechanism is desired), the tool may record the raw steps (e.g., mouse button clicks, keystrokes) that a user performs interacting with the application. The tool may use a special recording mechanism in various circumstances. These circumstances include recording of activities described in an application-specific internal macro language (recorded by the application using its own internal recording mechanism) and recording of activities against a dynamic user interface. The tools described herein may record activity against various kinds of applications, operating system software, or other software.

U.S. Patent Application Serial No. aa/bbb,ccc, entitled "Recording/Playback Tools for UI-based Applications," filed June 15, 2004, the disclosure of which is hereby incorporated by reference, describes various recording and playback tools. Some of these tools use the UIA framework to record input based on a UI element's identification value ("UID"). UIA generates a composite ID that uniquely identifies a UI element in a GUI. The UID depends at least in part on location of the UI element in the user interface and its relationship with other UI elements. When a tool uses a UID to record and playback



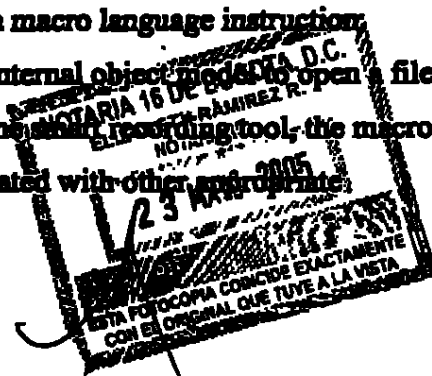
activities involving a specific UI element, changes to the element's location can cause playback to fail.

For example, suppose a user reconfigures a user interface by removing menu bar buttons. When a playback tool attempts to actuate the removed button, playback fails.

- 5 As another example, often when an application is upgraded features are moved and/or renamed to accommodate new features. For instance, suppose in one version an "Open File" feature is located as the third option on a "File" menu. In a subsequent release, the "Open File" feature is moved to accommodate a "New File" feature, and "Open File" becomes the fourth item listed on the "File" menu. Macros based on the old user
10 interface may break because the UI element moved and/or the UID has changed.

- In some implementations, a system-wide macro and UI recording tool uses smart recording to address these kinds of problems. An application records user actions in the application's internal macro language. After recording, the recorded steps expressed in the application's internal macro language are passed as opaque recorded step tokens from
15 the application to the macro and UI recording tool. The system-wide macro and UI recording tool may then further encapsulate the recorded step tokens and/or associate the recorded step tokens with marking information. In some implementations, each macro instruction is converted into a separate recorded step token. Alternatively, a series of related macro language instructions are wrapped into one recorded step token. Those
20 tokens maintain the temporal sequencing of the underlying macro language instructions so as to permit them to be played back by the application as they were recorded, while also allowing for interoperation with the generalized, system-wide macro and UI recording tool. After receiving the recorded step token, the tool optionally generates code that reflects the recorded steps. Subsequently, the code may be played back to
25 recreate the recorded steps or, alternatively, the code may be edited for testing, troubleshooting, or other purposes.

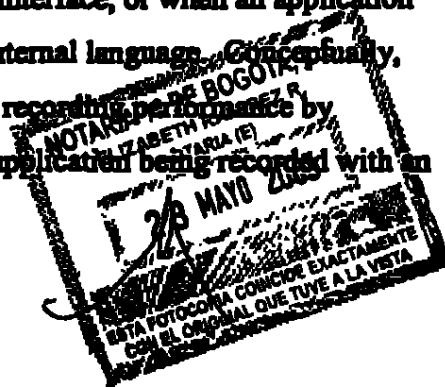
- For example, suppose a word processor has a macro language instruction "Open(filename)" that hooks into the application's internal object model to open a file with the name "file name." During recording with the smart recording tool, the macro
30 instruction is passed as a recorded step token, associated with other appropriate



information, and stored for playback, at which time the underlying instruction, "Open(filename)," is passed as a recorded step token from the smart recording tool back to the application. This permits applications to use their own native recording environments to record and playback UI activity within a system-wide UI recording and playback framework. A system-wide macro may be written that is not subject to the problems associated with menu customizations and changes in user interfaces associated with software upgrades, so long as the underlying macro language remains compatible with prior versions of the software. Moreover, the system-wide recording framework provides a unified recording and playback interface. It also simplifies the interleaving of macros and other UI recordings for different applications in combined macro and UI recording processes.

As part of the system-wide UI recording framework, when an application does not use an internal macro recorder or other application-specific UI recorder, a different recording mechanism may be used. For example, a UIA-based recording system is used, as described in the U.S. patent application entitled "Recording/Playback Tools for UI-based Applications," which works well for recording user interface activity against applications having a static set of menus. Alternatively, if an application does not support operation according the UI Automation architecture, the system-wide UI recording framework RPF may simply record the raw steps a user performs as he or she interacts with the UI-based application. In some cases, the various recording techniques are used in combination. For example in some test cases the raw user input, the UIA-based data, and internal macro recorder data are all recorded.

Recording the raw steps (e.g., mouse clicks, keystrokes) that a user performs interacting with a UI-based application is relatively simple; the challenges are in how to represent and process this information when an application has a configurable user interface (e.g., one which allows the menus and menu items to be rearranged at will), when an application is upgraded modifying the user interface, or when an application processes macro or UI recording information in an internal language. Conceptually, some techniques and tools described herein improve recording performance by integrating the internal recording capabilities of the application being recorded with an



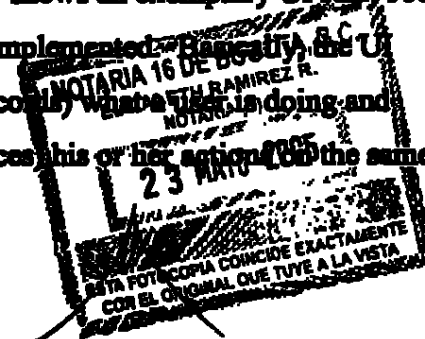
abstract, user-friendly interface through which a user may convert recorded steps and input into playback code. As noted above, applications with their own internal recording capabilities may invoke their internal recorder to map a user's actions to the application's internal object model, thus solving the "upgrade lock-out" and menu reconfiguration problems caused by simple menu/menu-item recording.

In some implementations, application-specific macro or UI recording data is collected and then converted into recorded step tokens using tokenizing techniques. The techniques and tools create opaque tokens defining user interaction with an application. Once the recorded step tokens have been generated, other recorded actions (e.g., UI events also in tokens) may be converted into programming code for playback on the same or another computer, potentially with application-generated recorded step tokens still embedded within the programming code.

In various common situations, the techniques and tools described herein allow a system-wide macro or UI recording tool to interoperate with various different kinds of software programs (e.g., UIA compatible applications, applications with internal macro recorders, applications that are non-UIA-compliant) and various kinds of user interfaces (e.g., static, dynamic). These techniques and tools make recording reliable and robust by using an application's own internal recording capabilities when appropriate. Moreover, the tools and techniques provide users with a consistent macro or UI recording interface. The techniques and tools can be implemented in various ways, and may be used in combination or separately.

L UI Recording and Playback Framework

A UI Recording and Playback Framework ("UI RPF") is an extensible framework designed to facilitate the recording and playback of UI activity against applications or other software programs. Pluggable modules into the framework include filters, aggregators, tokenizers, and code generators. Figure 1 shows an exemplary UI RPF in which various techniques described herein may be implemented. The UI RPF 100 records (or works with other software that records) what a user is doing and reproduces (or works with other software that reproduces) this or her actions on the same

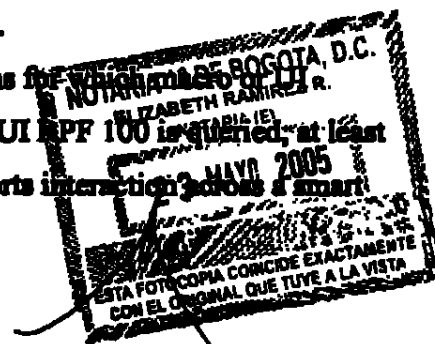


or another computer. UI RPF 100 is a cross-application (e.g., system level) framework that can be used with multiple applications and in many configurations, and it is designed to facilitate recording and playback of various different kinds of user activity. It is a foundation upon which recording and playback tools can be built. The UI RPF 100 may use services provided by UI system architecture software 120, such as Microsoft UI Automation, to facilitate UI recording, but this is not a requirement of the framework 100.

Figure 1 shows various modules associated with the UI RPF 100. The UI RPF 100 interacts with the UI system architecture software 120, which converts recorded UI input data such as mouse clicks, keystrokes, and input from applications 125, 126 into data that is more meaningful for UI recording and playback. The framework 100 receives recorded steps from applications such as a text editor 125 and calculator 126 through the UI system architecture software 120, possibly passed as tokens encapsulating system user interface actions. In some implementations, the UI RPF 100 employs a UI recording and playback mechanism described in the U.S. patent application entitled, "Recording/Playback Tools for UI-based Applications," to record and playback UI activity received through the UI system architecture software 120. Alternatively, the UI RPF 100 uses a different UI recording and playback mechanism.

The modules of the UI RPF 100 also include a smart recording and playback module 110 for interoperating with applications that have their own native macro recorder. Bypassing the UI system architecture software 120, the framework 100 receives recorded steps or other macro information in recorded step tokens from applications such as word processor 115, spreadsheet 116, and photo editor 117. Applications 115, 116, 117 have their own internal recording capabilities to map a user's input to actions in the application's native recording environment, then the applications 115, 116, 117 pass the internally recorded actions as tokens to the framework 100 through the smart UI recording and playback module 110.

The UI RPF 100 maintains a list of the applications for which macro recording activity may be recorded. A new application seen by the UI RPF 100 is queried, at least the first time the application is launched, to see if it supports interaction through a smart



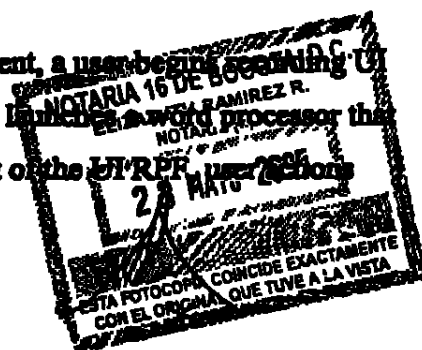
recording interface. If the application does support interaction across the smart interface, then user actions may be recorded according to the application's recording capabilities and passed back and forth through the interface. Alternatively, user actions are recorded through both the application's internal recording mechanism (with data passing through the smart recording interface) and through the UI system architecture software 120, where both options are possible. Or, even if both options are possible, a user, application, or system-wide tool may decide to use either the application's internal recording mechanism or recording through the UI system architecture software 120, but not both. Or, the user, application, or system-wide tool may decide to record the raw steps (e.g., mouse movements, mouse button clicks, keystrokes) associated with the user actions, either separately or in combination with the other recorded data.

When user activity recorded through an application's internal recording mechanism is in turn recorded by the UI RPF through the smart interface, the application converts the recorded user activity events into tokens. The contents of these tokens are defined in Visual Basic for Applications ("VBA") or in some other language or object model used by the application. Ultimately, the information stored in the "payloads" of these tokens is not interpreted by the UI RPF 100, but only by the respective applications for which the information is in a native format. The UI RPF 100 may further tokenize recorded step tokens and associate the data with other relevant data such as the name of the application.

In the system-wide tool, recorded step token data may be concatenated with recorded step token data obtained from other applications with native recorders, or with recorded data obtained through the UI system architecture software 120, so as to implement system level recording and playback.

In addition to recording macro and UI data, the framework 100 may produce playback code that replicates the recorded activity and/or encapsulates data that allows other software to replicate the recorded activity.

For example, in a Microsoft Windows environment, a user begins recording UI activity, then clicks on the Windows "Start" button, and launches a word processor that works with the smart recording interface. In the context of the UI RPF user actions



415

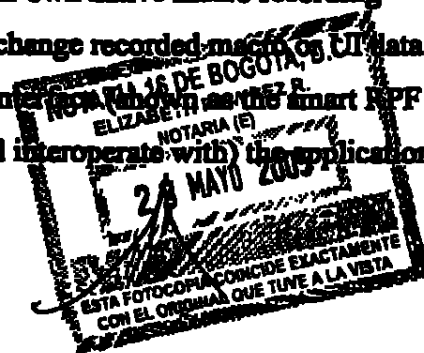
including the mouse click on the "Start" button are recorded. When the word processor is launched, the UI RPF checks to see if the word processor is compatible with the smart recording interface. If it is, the UI RPF invokes the proper tools within the word processor itself to record user actions performed against it. If the word processor is also enabled to work with the UI system architecture software, the tool may also record user actions (UI Automation events) in conjunction with the UI system architecture software. Additionally, the tool may also record the raw steps (e.g., mouse button clicks, keystrokes) associated with user actions, which may be particularly relevant if the word processor does not support the smart recording interface and is not enabled to work with the UI system architecture software. Thus, the UI RPF may use alternative mechanisms for recording and playback.

This framework 100 may include elements of software and/or hardware. The relationships shown between modules in Figure 1 indicate the main flow of information; other relationships are not shown for the sake of simplicity. Depending on implementation, modules can be added, omitted, split into multiple modules, combined with other modules, and/or replaced with like modules. Alternatively, a framework with different modules and/or other configurations of modules performs one or more of the UI recording and playback techniques described herein.

20 II. Smart Recording

Figure 2 shows a system-wide UI recording tool 220 in conjunction with which various techniques described herein may be implemented. The UI RPF of Figure 1 includes the recording tool 220. Alternatively, the recording tool 220 is used as part of another UI framework.

25 The recording tool 220 is system-wide in that it records UI system architecture information for UI recording with certain applications, but also interoperates with applications that record user interface activities in their own native macro recording environments. Such native recording applications exchange recorded macro or UI data with the tool 220 across the special native recording interface (shown as the smart RPF API in Figure 2). This allows the tool 220 to use (and interoperate with) the application's



own recording capabilities, which may thus allow UI recording for dynamic user interfaces and/or UI recording that is re-playable for changed user interfaces for different versions of an application.

The recording tool 220 controls the mechanism(s) to be used when recording macro or UI data. For example, depending on user input, user settings, or other criteria, the tool 220 decides whether to use native application recording, UI-system recording, or both for a particular application. The tool may also record the raw UI input from the user's mouse and keyboard, either separately or in combination with the other recording mechanisms. The tool 220 may concurrently record macro or UI data for each of multiple applications in use, or for selected applications among multiple applications in use. For example, if a user configures the recording tool 220 to only record data from a word processor program, as a user records the cutting and pasting of data from a spreadsheet program to the word processor, only the steps related to the word processor are recorded.

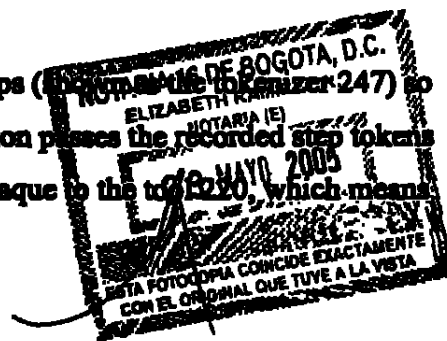
15

A. Native Recording Path

For native macro or UI recording, a user interacts with the application 240 using user input devices such as a mouse, a keyboard, a touch pad, touch screen, Braille screen, joystick, microphone, or other similar device. As the user interacts with the application 240, the user inputs are processed and recorded by the macro/UI recorder 245 as steps in the application's native macro recording environment, as expressed in a language such as VBA, PerfectScript, M4, TeX, or other macro language. The application's internal recording capabilities provide a robust framework for acquiring data that may be predictably and reliably played back by the application. Recorded steps are only limited by the limitations of an application's native recording environment. For example, if an application permits a user to declare variables in a macro, those variables may be recorded and tokenized.

The application 240 tokenizes the recorded steps (the tokens are tokenized by the application 240) so they can be passed to the tool 220. Then, the application passes the recorded step tokens to the tool 220 across the interface. The tokens are opaque to the tool 220, which means

30



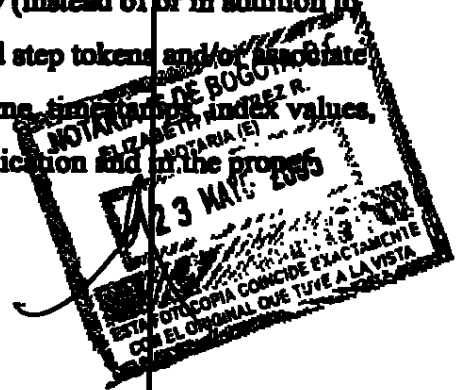
117

the tool 220 handles the recorded step tokens as discrete, opaque chunks of data whose contents are not interpreted by the tool 220. The application 240, on the other hand, may interpret and react to the contents of the recorded step tokens. The contents are in a native format for the application 240. For example, the application 240 encapsulates as a token recorded macro instructions to open a file exactly as they were recorded (e.g., as binary data, a text command, or a DLL instruction).

The application 240 converts each discrete recorded step into a token, or the application 240 groups multiple recorded steps as a single token. As for the mechanics of the tokenization, the application 240 may group recorded step data into one or more fields of a data structure for a token, which is then passed by copy or reference to the tool 220. Or, the application 240 may pass recorded step data as parameters of a method call, which the tool 220 receives and handles as opaque data. The application 240 may use an XML rules sheet (which defines a set of rules for tokenizing recorded step data) to reformat or reorganize the recorded step data into tokens. Or, the application 240 uses some other mechanism for tokenization. Conceptually, the tokenization may be viewed as placing the recorded step data in a token data structure, or wrapping the recorded step data with token information. This "hides" the macro language instructions from the tool, creating a token that is macro language independent but whose contents are recognizable in the appropriate native recording environment.

The application 240 may prepend, append, or otherwise add marking data to the recorded step when tokenizing the data. The marking data provides relevant details for playback, for example, application information (e.g., the name of the application and its version), a timestamp to indicate an order, privileges information (e.g., what rights are needed to play the step), etc.

The tool 220 receives tokens (shown as the token acceptor 235) passed from one or more applications over the special native recording interface. While the tool 220 does not interpret the contents of the recorded step tokens, it may (instead of or in addition to the application) add additional layers of data to the recorded step tokens and/or associate the tokens with marking information such as application name, time stamps, index values, etc. so that the steps may be played back by the correct application and in the proper



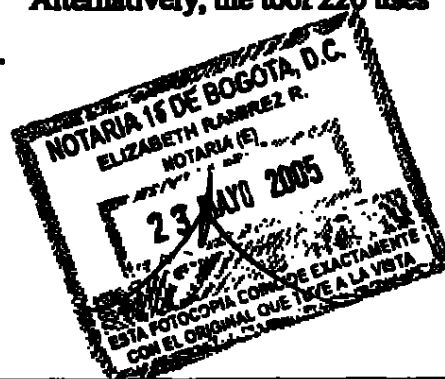
sequence. In at least this sense, the tool 220 may be considered to also perform tokenization.

The application 240 supports the native recording interface in that it formats information for passing to the tool 220 across the interface, makes the appropriate calls across the interface, implements callback methods for the tool 220 to call, etc. The tool 220 supports the interface in the sense that it implements methods for applications to call, handles information passed in by applications as expected, calls callback methods of applications, etc. The exact call and processing mechanisms used for the interface are implementation dependent.

B. UI System Recording Path

The application 240 with the native macro recorder and/or another application 260 may support interaction with the system recording tool 220 through the UI system architecture software 120. For UI system architecture recording, various components 225 of the recording tool 220 collect (a) user input data, (b) monitor other UI activity (such as focus change, selection change, window creation, etc.), (c) track information about the environment such as running applications, OS version, localization information, default UI language, current UI language, hardware devices and configurations, security modes, and/or other data that may help describe or recreate the proper environment for playback, (d) resolve the user input data to user interface elements affected by the user input, and (e) express the user activities as logical recording events. These logical recording events help provide readable UI recording output and reliable UI recording playback. Logical recording events may be considered as a type of UI recording token, but unlike the opaque recorded step tokens with native recording data, the tool 220 interprets the logical recording event tokens for viewing, editing, conversion to code, etc.

In some implementations, the tool 220 performs UI system architecture recording with filters and aggregators as described in the U.S. patent application entitled "Recording/Playback Tools for UI-based Applications." Alternatively, the tool 220 uses another technique for UI system architecture recording.



119

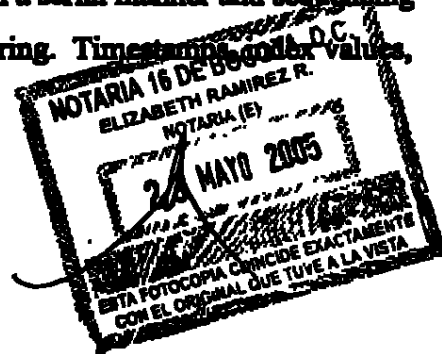
The tool 220 may display recorded UI system architecture steps in a viewer such as a debugger window, source code browser, or text editor so users may read and browse the data for errors. Typically, the tool 220 does not display information about the recorded step tokens with native recording data, as the contents of such tokens are opaque to the tool 220. When the recorded step tokens are associated with marking information such as application name, timestamps, action labels or other annotations from the application or user, the tool 220 may display such information. The data may similarly be saved to a file such as a log file for later analysis.

10 C. Other User Input

Various components 225 of the recording tool 220 may also collect raw user input data and similarly monitor other UI activity, track information about the environment such as running applications, etc. Recording these events helps in situations where an application is not compatible with the smart recording interface or the UI system architecture. These recorded actions may also be displayed in a viewer so users may read and browse the data for errors. Typically, the data from the raw user actions is less readable and reliable for playback than the other types of recorded tokens as explained above in the background. However, when the other recording techniques are inapplicable, this provides a mechanism for recording non-UIA and non-native recorder application data in the system-wide UI RPF. The data recorded from raw user actions may similarly be saved to a file such as a log file for later analysis.

D. Playback Code

In some implementations, the tool 220 combines tokens from different applications and recording modes in a single file by interleaving the data. For example, recorded step tokens for native recording actions for different native recorders are combined into one input stream, along with tokens for UI system architecture actions and any other user input. Typically, user input is received in a serial manner and sequencing of tokens in this manner a stream maintains proper ordering. Timestamps, context values,



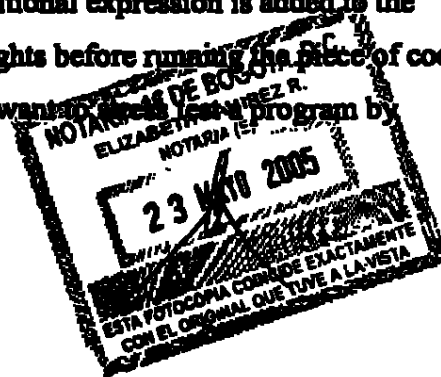
and application information may mark tokens to help maintain proper ordering. Or, the tool 220 creates separate files for different applications and recording modes.

For UI system action tokens, the tool 220 uses a code generator 230 to generate playback code 250 from the tokens. This can be done automatically or when the user selects a "Generate Code" option in the tool 220. The playback code 250 may be output in a variety of formats, including C#, XML, C++, JAVA, or any other programming, scripting, macro, or document-oriented language, or other commands or instructions for a playback interface or other mechanism to drive UI-based applications. For example, the code generator 230 maps tokens for UI system actions to corresponding instructions in the playback code 250. For additional details on code generation in a UI system architecture environment, see the U.S. patent application entitled, "Recording/Playback Tools for UI-based Applications," filed June 15, 2004.

The tool 220 optionally puts the recorded step tokens for native recording actions into playback code 250 such as a script file that maintains the recorded step tokens in correct temporal sequence. Or, the tool 220 uses a code generator 230 to generate playback code 250 that includes the recorded step tokens. Notably, the recorded step tokens that encapsulate native recording data are opaque to the tool 220, so such tokens are typically passed through to the playback code 250, and are not interpreted by the tool 220 for conversion to other expressions.

The playback code 250 may be viewed, saved, edited, modified, added to, deleted from, compiled, or otherwise used prior to playback, with the expected limitations on the extent to which the tool may view, modify, etc. opaque recorded step tokens. Being able to edit the playback code 250 provides users with a convenient and powerful mechanism for creating macros, generating automatic test cases, automating steps in a program, etc. The generated code may be edited to add conditional expressions, to change variables, to add loops, and other programming instructions to the code. For example, on playback, suppose a piece of playback code alters a system file and requires that a user to be logged on with Administrator rights. Accordingly, a conditional expression is added to the playback code, which checks for the appropriate rights before running the piece of code.

As another example, suppose software developers want to test a program by



repeating the same series of instructions over and over again, so, a "for loop" is added to the playback code, which causes the playback code to be repeated a certain number of times.

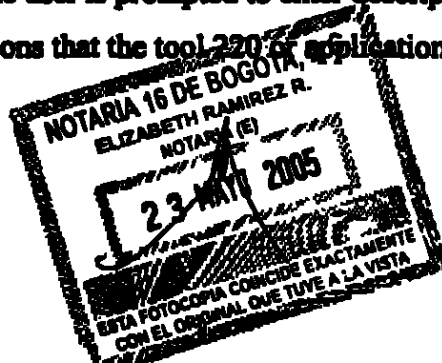
The playback code 250 may then be used to manage playback of the recorded activity, as described below. The code 250 may be played back on the same computer or, alternatively, on a different computer(s).

E. Tool Architecture and Uses

In general, the various modules of the tool 220 are pluggable, which means they may be replaced with other modules with similar functionality for different applications. For example, the code generator 230 in the tool 220 is pluggable, so a developer can reuse some parts of the tool 220 but change the way code is generated by switching code generators. The tool 220 is thus extensible; its functionality may be altered for different applications. In some implementations, the tool 220 presents a menu of options for pluggable modules, thus allowing a user to control loading of appropriate modules for the functionality desired by the user.

For example, in a "heavyweight" recording mode, the tool 220 collects detailed information about UI activities and comprehensively logs the information. This provides more reliable playback but is more likely to perturb execution of the software against which activity is recorded. Or, in a "lightweight" recording mode, the tool 220 collects less information and, therefore, is less likely to perturb execution of the software against which activity is recorded. Yet, this may still provide enough information for playback in the same environment, for showing user activity to support personnel, or for other purposes.

Similarly, the tool 220 may record information in an interactive mode in which UI elements are highlighted by a bounding rectangle or flashing to show what is recorded. Other flashing feedback may be used to show the user the applications for which UI data is being recorded. Or, in the interactive mode, the user is prompted to enter descriptive annotation information for each action or for actions that the tool 220 or application 240



are unable to resolve. Alternatively, the tool 220 may operate in the background to record UI activity without disturbing the user.

The tool 220 is responsible for loading other pluggable modules as needed (e.g., modules for collecting data, tokenizing data, generating code, etc.), enforcing security (e.g., checking for privileges), spawning recording threads, synchronizing activities between modules, building and maintaining queues used for recording, logging errors (e.g., to a local file or optionally to a network service or server), and collecting information about the recordings (e.g., OS version, localization information, hardware configuration, etc.).

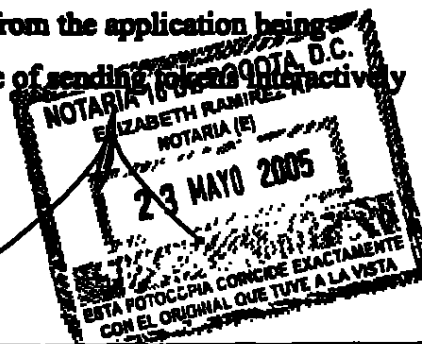
The relationships shown between modules in Figure 2 indicate the main flow of information; other relationships are not shown for the sake of simplicity. Depending on implementation, modules can be added, omitted, split into multiple modules, combined with other modules, and/or replaced with like modules. Alternatively, a tool with different modules and/or other configurations of modules performs one or more of the recording techniques described herein.

The user interface of the tool 220 may be a VCR-like interface positioned around a recording window, docked to the top or bottom of a computer screen, or minimized to a task bar. The interface includes buttons for features such as "Record," "Pause," "Stop," and "Play," as well as an area for presenting description of recorded steps when such description is available. For examples of such interfaces, see the U.S. patent application entitled "Recording/Playback Tools for UI-based Applications."

F. Recording Technique, Timing of Recording

There are two modes for transferring the tokens to a tool 220 during recording: batch mode and interactive mode. In general, in batch mode, recorded step tokens are collected at the application and delivered to the tool 220 when requested. These requests to retrieve the collected tokens may occur when focus has changed away from the recording application, and may also occur periodically to distribute the recording load.

In interactive mode, in contrast, tokens are sent from the application being recorded to the tool 220 as they occur. One disadvantage of sending tokens interactively



is that it increases the cross-process loading on the CPU. In a particular situation, either batch or interactive mode may be preferable to the other, depending on the application being recorded, the amount of data being recorded, and how quickly recorded step tokens are being generated.

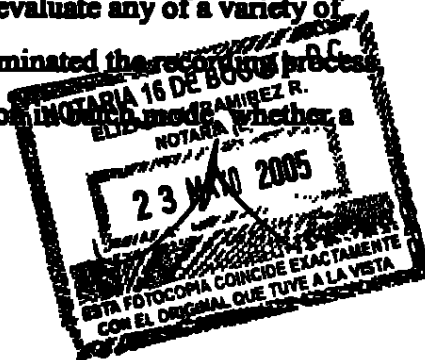
5 Figure 3 shows a technique 300 for recording application-native macro activity with a system-wide UI recording tool. A tool and application such as the recording tool 220 and application 240 shown in Figure 2 perform the technique 300. Alternatively, another tool and/or application perform the technique 300.

10 The tool checks 310 to see if the application uses a smart native recording interface. Some applications do not support interaction over the interface. Other applications might support interaction over the interface, but the user or tool determines not to use native recording. For example, when a user first launches an application the tool checks to see if the application supports interaction over the smart native recording interface, and if so, adds the name of the application to a list of applications that support
15 interaction over the interface. Or, the tool checks to see if an application supports the smart native recording interface at the time recording begins. If an application does not support interaction over the interface, the tool may add the application to a list of non-supported applications

20 If the smart native recording interface is not used, the tool uses 315 another recorder to record UI activity. For example, the tool uses a UIA software recorder for a UIA-enabled application, or alternatively, a raw UI input recorder for non-UIA-enabled applications. While Figure 3 shows alternative paths of recording, the tool may instead use both the native recording and the other recording concurrently.

25 When the smart native recording interface is used, the application receives user input and records 320 macro data. Thus, the application's native recorder records steps performed by the user against the application

30 The process of recording macro data continues until the application or tool determines 325 it should not continue. This check may evaluate any of a variety of conditions including whether the user has selectively terminated the recording process, whether the tool has requested tokens from the application, or whether a



counter or timer has expired in batch or interactive mode, whether resources are available for tokenization, whether a designated UI element has been accessed, whether a specified number of recorded steps reached, etc.

The application then tokenizes 330 the recorded macro data, producing one or more tokens that are opaque to the tool. For example, the application uses a tokenization mechanism as described above and/or adds marking information as described above. Alternatively, the tool performs some or all of the tokenization or information addition.

The tool or application evaluates 335 whether to continue with the macro data recording. Again, this check may evaluate any of a variety of conditions, including whether the user has selectively terminated the recording process, whether tokenization succeeded or annotation information should be solicited from the user, etc.

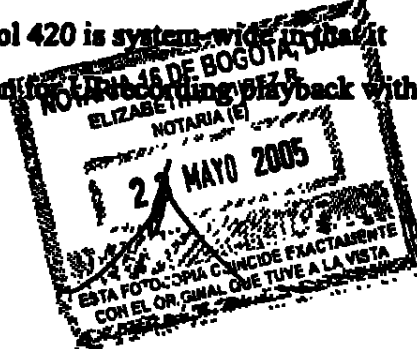
The tool optionally converts 340 the tokens it receives to code such as a script in which the tokens are embedded. For example, the tool uses a code generation mechanism as described above.

Alternatively, various stages of the technique 300 are separately or in various combinations performed in conjunction with other stages. For example, while Figure 3 shows the tokenizing 330 and conversion 340 occurring after macro data recording 320, alternatively one or both of these operations are performed concurrently with the recording 320.

III. Smart Playback

Figure 4 shows a system-wide UI playback tool 420 in conjunction with which various techniques described herein may be implemented. The UI RPF of Figure 1 includes the playback tool 420. Alternatively, the playback tool 420 is used as part of another UI framework.

The playback tool replicates UI activity by receiving playback code and driving applications through UI system services, by passing tokens to applications for interpretation in their own native recording environment, or through playing back raw steps associated with the UI activity. The playback tool 420 is system-wide in that it handles playback of UI system architecture information for recording playback with



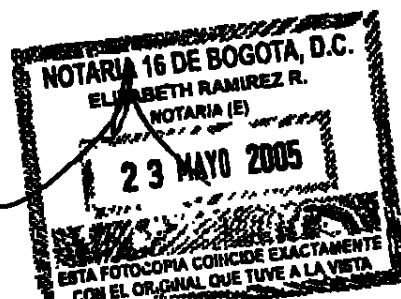
125
certain applications, but also interoperates with applications that play back user interface activities in their own native macro recorder environments. Such native recorder applications exchange recorded macro or UI data with the tool 420 across a special native playback interface (shown as the smart RPF API in Figure 4). This allows the tool 420 to control (and interoperate with) the application's own recording capabilities, which may thus allow UI playback for dynamic user interfaces and/or UI playback for changed user interfaces for different versions of an application.

The playback tool 420 controls the path(s) to be used when playing back macro or UI data. For example, depending on what data have been recorded, the tool 420 decides whether to use native application recorder playback, UI system playback, or both. Alternatively, the tool may also playback the raw recorded steps. The tool 420 may concurrently play back macro or UI data for each of multiple applications in use, or for selected applications among multiple applications in use.

15 A. Reader

As input, the playback tool 420 receives playback code 450 from a code generator such as the code generator 230 as illustrated in Figure 2 or elsewhere. The playback code 450 may be in a variety of formats, including C#, XML, C++, JAVA, or any other programming, scripting, macro, or document-oriented language, or other commands or instructions, for a playback interface or other mechanism to drive UI-based applications for UI system actions. Recorded step tokens for native recording actions may be included in code 450 such as a script file (which may be routed directly to the token passer 435) or embedded in other code 450.

In some implementations, the playback tool 420 receives a single file that combines code and tokens from different applications and recording modes by interleaving the data. For example, recorded step tokens for native recording actions for different native recorders are combined into one stream, along with code for UI system architecture actions and any other recorded user input. Or, the tool 420 receives separate files for different applications and recording modes.



126

nb

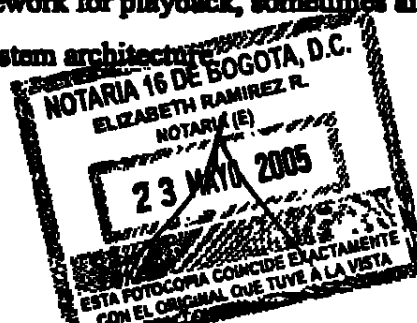
In the playback tool 420, the reader 410 receives playback code 450 and verifies the correctness of the code 450. For example, the playback tool receives a file containing playback code and attempts to verify that the file is internally consistent and consistent with the current status of the system. If any inconsistencies are found, the user is notified, and, in some implementations, the user is asked to manually complete the step. Alternatively, playback is terminated.

For playback according to the UI system architecture playback path, the reader 410 passes the code 450 to the UI system architecture player 425 for programmatic control of applications as specified by the code 450. For playback according to the native recording playback path, the reader 410 passes recorded step tokens for native recorded data to the token passer 435.

B. Native Recorder Playback Path

For native macro or UI recorder playback, the playback tool 420 passes tokens (shown as token passer 435) from the tool 420 to the application 440 that includes the appropriate native recording environment. The tool 420 passes the tokens over the special native playback interface. Before doing so, the tool 420 may interpret additional layers of data that were added to the recorded step tokens, including marking information such as application name, timestamps, index values, etc., so that the steps are played back by the correct application and in the proper sequence.

Unlike other kinds of information in the code 450, recorded step tokens for native recording data are not directly interpreted by the playback tool 420. Instead, the tokens are passed as opaque chunks of data by the token passer 435 to the application 440 (shown as the token acceptor 442 component). The application 440 interprets the tokens, passing the macro recording data from the tokens to the macro player 445. In the application 440, recorded macro data is interpreted in the application's native macro recording environment. The macro data is expressed in a language such as VBA, PerfectScript, M4, TeX, or another macro language. An application's internal recording capabilities provide a robust and reliable framework for playback, sometimes allowing playback that is not possible through the UI system architecture.



427

127

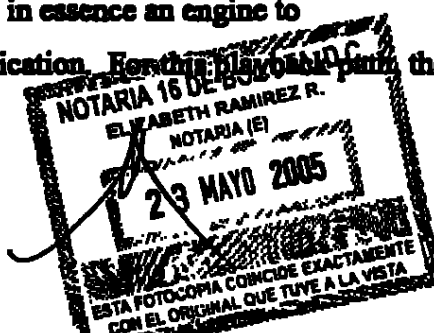
The application 440 extracts the recorded steps to prepare the recorded steps for playback. The application 440 may extract one recorded step action from a single token, or the application 440 may extract multiple recorded step actions from a single token. As for the mechanics of the reverse tokenization, the application 440 may read recorded step action data from one or more fields of a data structure for a token, which was passed by copy or reference from the tool 420. Or, the application 440 may unpack recorded step action data passed as parameters of a method call. Or, the application 440 uses some other mechanism to extract recorded step actions from tokens. Conceptually, the extraction may be viewed as reading the recorded step action data from a token data structure, or unwrapping the recorded step action data from token information. This "reveals" the macro language instructions, which are recognizable in the appropriate native recording environment.

The application 440 may also process marking data that was prepended, appended, or otherwise added to the recorded step action data when tokenizing the data. The marking data provides relevant details for playback, for example, application information (e.g., the name of the application and its version), a timestamp to indicate an order, privileges information (e.g., what rights are needed to play the step), etc.

The tool 420 supports the special native playback interface in the sense that it formats information for passing to the application 440 across the interface, makes the appropriate calls across the interface, implements callback methods for the application 440 to call, etc. The application 440 supports the interface in the sense that it implements methods for the tool 420 to call, handles information passed in by the tool as expected, calls callback methods of the tool 420, etc. The exact call and processing mechanisms used for the interface are implementation dependent.

C. UI System Playback Path

The application 440 with the native macro recorder playback and/or another application 460 may support playback according to the UI system architecture with the UI system architecture player 425. The player 425 is in essence an engine to programmatically control features of a UI-based application. For this playback path, the



123
playback tool 420 processes the code 450 with the player 425 to replicate the recorded steps and UI activity through programmatic interaction with the UI-based application.

In some implementations, the tool 420 performs UI system architecture playback with components as described in the U.S. patent application entitled "Recording/Playback
5 Tools for UI-based Applications," filed June 15, 2004. Alternatively, the tool 420 uses another technique for UI system architecture playback.

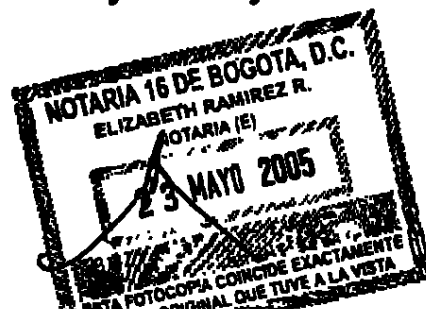
At the start of playback, a comparison may be made between the current conditions and the conditions under which the UI recording was made. If significant variations are detected, the user is informed, and offered the option to continue.

10 In some cases, the playback code is self-executing and requires no additional input or action to control. Alternatively, UI system actions are put into an internal playback queue, which allows the player module 425 to process each recorded step in order. The player 425 gets a recorded step from the queue, waits for the appropriate application to become available, waits for the process to be ready for input, executes the
15 recorded step, and reports the result of the step to the playback tool 420 or elsewhere. If any of those steps fail, a simple sleep function is called and the player 425 simply waits a certain period of time before attempting to reproduce the recorded step again. Or, when a step cannot be performed, the user is notified and asked to complete the step. This allows playback failures to be handled gracefully without crashing the playback tool or any other
20 associated applications.

Instead of relying on the player 425, some applications may also substitute their own playback tool to run generated code.

D. Tool Architecture and Uses

25 In general, the various modules of the tool 420 are pluggable, which means they may be replaced with other modules with similar functionality for different applications. The tool 420 is thus extensible; its functionality may be altered for different applications. The tool 420 may present a menu of options for pluggable modules, thus allowing a user to control loading of appropriate modules for the functionality desired by the user.



129

The playback tool provides various configurable options. The playback tool 420 may be configured to indicate how playback is to take place. For example, the playback tool may be configured to pause during playback when a "Pause" button is actuated. Or, the tool 420 pauses when user input is required to continue processing the playback code.

5 The playback tool may be configured to perform playback as fast as the computer system running the code can "Paint." Alternatively, playback occurs as fast as possible without waiting for Paint to complete. In some implementations, multiple playback series are run simultaneously against separate applications.

To improve the security of playback, the recording tool 220 of Figure 2 may

10 encrypt playback code and/or the tokens embedded therein. If so, the playback tool 420 decrypts the code and/or tokens. Other security measures include requiring that users re-enter user names and passwords on playback, not logging such data, checking the security mode of the playback code against the security mode of the user (e.g., requiring administrator status for users trying to play back administrator-level scripts), and only

15 allowing buffer playback (playback from code and/or tokens not put in a file) when it can be verified that the buffer contents were recorded by the current user, in the current session.

The tool 420 is responsible for loading other pluggable modules as needed, enforcing security (e.g., checking for privileges), spawning playback threads,

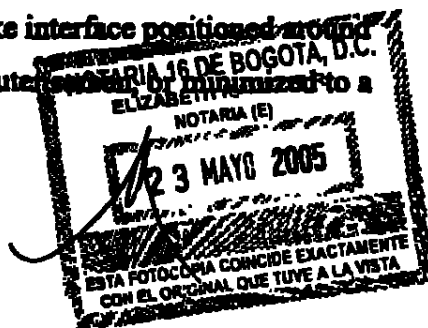
20 synchronizing activities between modules, building and maintaining queues used for playback, logging errors (e.g., to a local file or optionally to a network server or service), and collecting information about playback.

The relationships shown between modules in Figure 4 indicate the main flow of information; other relationships are not shown for the sake of simplicity. Depending on

25 implementation, modules can be added, omitted, split into multiple modules, combined with other modules, and/or replaced with like modules. Alternatively, a tool with different modules and/or other configurations of modules performs one or more of the playback techniques described herein.

The user interface of the tool 420 may be a VCR-like interface positioned around

30 a playback window, docked to the top or bottom of a computer monitor or minimized to a



180

task bar. The interface includes buttons for features such as "Pause," "Stop," and "Play," as well as an area for presenting description of the steps being played back when such a description is available. For examples of such interfaces, see the U.S. patent application entitled "Recording/Playback Tools for UI-based Applications."

5

E. Playback Techniques

Figure 5 shows a technique 500 for managing playback of application-native macro activity with a system-wide UI playback tool. A tool and application such as the playback tool 420 and application 440 shown in Figure 4 perform the technique 500.

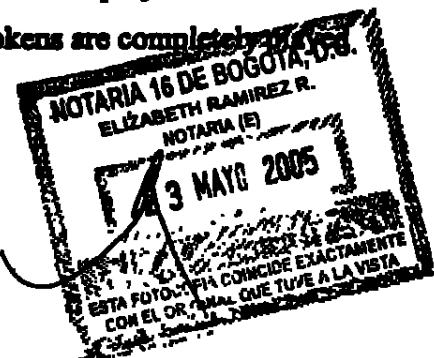
10 Alternatively, another tool and/or application perform the technique 500.

The tool receives 510 code with recorded step tokens. For example, the code is a script file including the tokens. Or, the code is programming language code having the tokens embedded therein.

15 The tool then verifies 520 the code. The tool may verify the correctness, ordering, and internal consistency of the tokens, or verify that the tool and application are ready for correct playback. In doing so, the tool may consider marking information added to tokens for the sake of identification or sequencing. For example, the tool uses a verification mechanism described above. If a recorded action cannot be verified, the user may be prompted to terminate playback or, alternatively, perform an action.

20 The tool then passes 540 the recorded step tokens to the appropriate application. The application processes 550 the tokens, extracting recorded macro data, and plays back the recorded native macro actions in the application's native recording playback environment to replicate the steps performed by a user against the application. Processing the native recorded actions typically involves replicating recorded UI activity
25 as smoothly as possible with little or no interference by a user. Thus, a system UI recorder tool may effectively manage and interoperate with the application's internal recording environment with a high degree of reliability.

The tool or application determines 560 whether to continue playback. For example, the tool and application finish when the code or tokens are completely played.



131

back, when the user terminates playback, or when playback ends due to an error, or the tool and application wait when playback stalls to solicit data entry from the user.

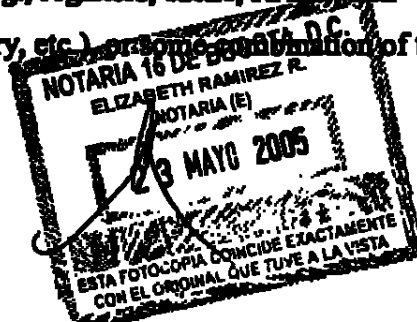
Alternatively, various stages of the technique 500 are separately or in various combinations performed in conjunction with other stages. For example, while Figure 5 shows the verification 520 and passing 540 occurring before token processing 550, alternatively one or both of these operations are performed concurrently with the processing 550 for tokens in a playback pipeline.

IV. Computing Environment

The above described tools for macro and UI recording and playback for diverse GUI-based software (e.g., the tool 200 of Figure 2) and macro and UI recording techniques can be implemented on any of a variety of computing devices and environments, including computers of various form factors (personal, workstation, server, handheld, laptop, tablet, or other mobile), distributed computing networks, and Web services, as a few general examples. The macro and UI recording tools and techniques can be implemented in hardware circuitry, as well as in macro recording, processing and viewing software 980 executing within a computer or other computing environment, such as the one shown in Figure 6.

Figure 6 illustrates a generalized example of a suitable computing environment 600 in which the described techniques can be implemented. The computing environment 600 is not intended to suggest any limitation as to scope of use or functionality of the tools and techniques described herein, as they may be implemented in diverse general-purpose or special-purpose computing environments.

With reference to Figure 6, the computing environment 600 includes at least one processing unit 610 and memory 620. In Figure 6, this most basic configuration 630 is included within a dashed line. The processing unit 610 executes computer-executable instructions and may be a real or a virtual processor. In a multi-processing system, multiple processing units execute computer-executable instructions to increase processing power. The memory 620 may be volatile memory (e.g., registers, cache, RAM), non-volatile memory (e.g., ROM, EEPROM, flash memory, etc.), or some combination of the



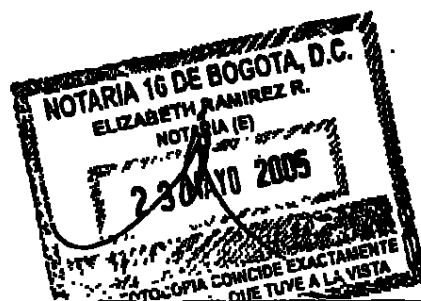
two. The memory 620 stores software 680 implementing the smart recording tools and techniques.

A computing environment may have additional features. For example, the computing environment 600 includes storage 640, one or more input devices 650, one or more output devices 660, and one or more communication connections 670. An interconnection mechanism (not shown) such as a bus, controller, or network interconnects the components of the computing environment 600. Typically, operating system software (not shown) provides an operating environment for other software executing in the computing environment 600, and coordinates activities of the components of the computing environment 600.

The storage 640 may be removable or non-removable, and includes magnetic disks, magnetic tapes or cassettes, CD-ROMs, CD-RWs, DVDs, or any other medium which can be used to store information and which can be accessed within the computing environment 600. The storage 640 stores instructions for the macro and UI recording and playback software 680.

The input device(s) 650 (e.g., for devices operating as a control point in the device connectivity architecture) may be a touch input device such as a keyboard, mouse, pen, or trackball, a voice input device, a scanning device, or another device that provides input to the computing environment 600. For audio, the input device(s) 650 may be a sound card or similar device that accepts audio input in analog or digital form, or a CD-ROM reader that provides audio samples to the computing environment. The output device(s) 660 may be a display, printer, speaker, CD-writer, or another device that provides output from the computing environment 600.

The communication connection(s) 670 enable communication over a communication medium to another computing entity. The communication medium conveys information such as computer-executable instructions, audio/video or other media information, or other data in a modulated data signal. A modulated data signal is 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



23

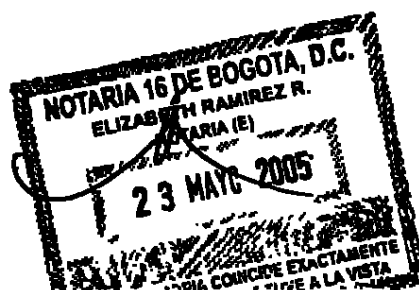
media include wired or wireless techniques implemented with an electrical, optical, RF, infrared, acoustic, or other carrier.

The macro and UI recording and playback techniques herein can be described in the general context of computer-readable media. Computer-readable media are any
5 available media that can be accessed within a computing environment. By way of example, and not limitation, with the computing environment 600, computer-readable media include memory 620, storage 640, communication media, and combinations of any of the above.

The techniques herein can be described in the general context of computer-
10 executable instructions, such as those included in program modules, being executed in a computing environment on a target real or virtual processor. Generally, program modules include routines, programs, libraries, objects, classes, components, data structures, etc. that perform particular tasks or implement particular abstract data types. The functionality of the program modules may be combined or split between program
15 modules as desired in various embodiments. Computer-executable instructions for program modules may be executed within a local or distributed computing environment.

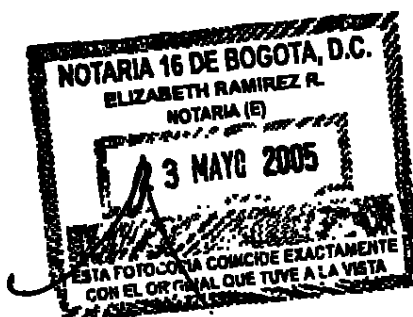
For the sake of presentation, the detailed description uses terms like "determine," "generate," "adjust," and "apply" to describe computer operations in a computing
20 environment. These terms are high-level abstractions for operations performed by a computer, and should not be confused with acts performed by a human being. The actual computer operations corresponding to these terms vary depending on implementation.

In view of the many possible embodiments to which the principles of my invention may be applied, I claim as my invention all such embodiments as may come within the scope and spirit of the following claims and equivalents thereto.



I claim:

1. A computer-readable medium storing computer-executable instructions for causing a computer system programmed thereby to perform a method comprising:
receiving a plurality of internal macro actions passed as one or more opaque
5 recorded step tokens from a graphical user interface-based application; and
recording the one or more opaque recorded step tokens with an external UI recorder.
2. The computer-readable medium of claim 1 wherein the plurality of internal
10 macro actions are generated by an internal macro recorder of the application and wherein the one or more opaque recorded step tokens are for handling by the external UI recorder to interoperate with the internal macro recorder of the application.
3. The computer-readable medium of claim 1 further comprising:
15 receiving a plurality of system user interface actions for user interactions against a second graphical user interface-based application;
generating one or more system user interface tokens for the plurality of system user interface actions; and
recording the one or more system user interface tokens with the external UI
20 recorder.
4. The computer-readable medium of claim 1 further comprising:
receiving a plurality of raw user actions for user interactions against a second
graphical user interface-based application;
25 recording the plurality of raw user actions with the external UI recorder.
5. The computer-readable medium of claim 1 further comprising associating the one or more opaque recorded step tokens with relevant data in the external UI recorder.



6. The computer-readable medium of claim 5 wherein the relevant data comprises application identification information for the application.

7. The computer-readable medium of claim 1 wherein the one or more opaque recorded step tokens are received by the external UI recorder in batch mode from the application.

8. The computer-readable medium of claim 1 wherein the one or more opaque recorded step tokens are received by the external UI recorder in interactive mode from the application.

9. The computer-readable medium of claim 1 wherein the external UI recorder interoperates with multiple different internal macro recorders for multiple different applications, wherein the multiple different applications include the graphical user interface-based application.

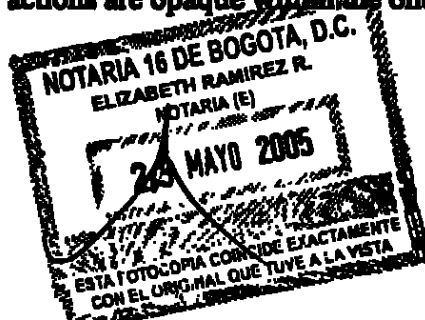
10. The computer-readable medium of claim 1 wherein the method further comprises:

passing the one or more opaque recorded step tokens from the external UI recorder back to the application; and
playing back the plurality of internal macro actions.

11. A computer-readable medium storing computer-executable instructions for causing a computer system programmed thereby to perform a method comprising:

receiving a set of one or more actions expressed in a first recording language used by a software program; and

representing the set of one or more actions as a set of one or more recorder tokens for a system UI recorder program, wherein the one or more recorder tokens encapsulate the one or more actions such that the one or more actions are opaque within the one or more recorder tokens.



12. The computer-readable medium of claim 11 wherein the software program generates the set of one or more recorder tokens.

5 13. The computer-readable medium of claim 11 wherein the system UI recorder program generates the set of one or more recorder tokens.

14. The computer-readable medium of claim 11 wherein the method further comprises:

10 receiving a second set of one or more actions expressed in a second recording language used by a second software program; and

representing the second set of or more actions as a second set of one or more recorder tokens for the system UI recorder program.

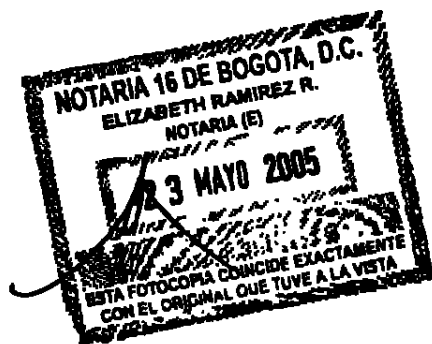
15 15. The computer-readable medium of claim 11 wherein the method further comprises:

receiving a second set of one or more actions expressed in a system UI recording language; and

20 representing the second set of one or more actions as a second set of one or more recorder tokens for the system UI recorder program.

16. The computer readable medium of claim 11 wherein the one or more recorder tokens are received by the system UI recorder program in batch mode from the software program.

25 17. The computer readable medium of claim 11 wherein the one or more recorder tokens are received by the system UI recorder program in interactive mode from the software program.



137

18. A system-wide UI recording tool for recording and playing back user interface activity, the tool comprising:

a module for collecting a first set of data associated with user interface activity against plural different applications, wherein the first set of data is expressed in any of plural native recording languages for the plural different applications;

a module for handling at least part of the collected first set of data as a set of tokens; and

a module for managing playback of the user interface activity by supplying the tokens to the respective plural different applications with which the first set of data is associated.

19. The tool of claim 18 further comprising a module for converting the set of tokens into output code

20. The tool of claim 18 wherein the modules are extensible.

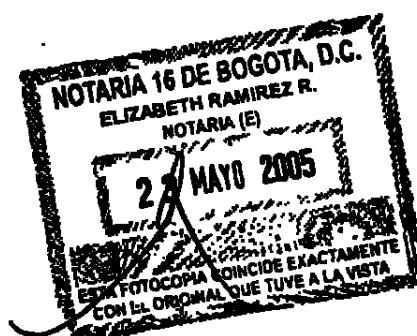
21. The tool of claim 18 further comprising:

a module for collecting a second set of data associated with second user interface activity, wherein the second set of data is expressed in a non-native, system UI recording language; and

a module for managing playback of the second user interface activity by playing back the second set of data.

22. The tool of claim 18 wherein the first set of data comprises macro language instructions.

23. The tool of claim of 18 wherein the collecting the first set of data is in batch mode.

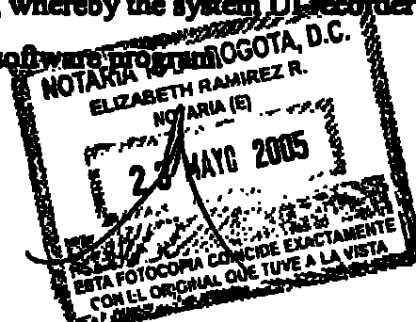


24. The tool of claim of 18 wherein the collecting the first set of data is in interactive mode.

25. A computer-readable medium storing computer-executable instructions for causing a computer system programmed thereby to perform a method comprising:
5 passing a set of one or more recorder tokens from a system UI recorder program to a software program for playback by an internal recorder of the software program; and
extracting a set of one or more internal recorder actions from the set of one or more recorder tokens, wherein the set of one or more internal recorder actions is
10 expressed in an internal recording language used by the software program, whereby the system UI recorder program interoperates with the internal recorder for the software program.

26. The computer-readable medium of claim 25 wherein the method further
15 comprises:
passing a second set of one or more recorder tokens from the system UI recorder program to a second software program for playback by a second internal recorder of the second software program; and
extracting a second set of one or more internal recorder actions from the second
20 set of one or more recorder tokens, wherein the second set of one or more internal recorder actions is expressed in a second internal recording language used by the second software program, whereby the system UI recorder program interoperates with the second internal recorder for the second software program.

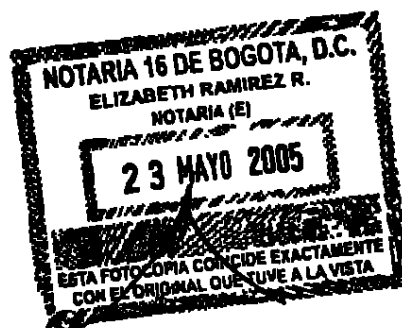
27. The computer-readable medium of claim 25 wherein the method further
25 comprises:
playing back a second set of one or more recorder tokens to programmatically control a second software program, wherein the second set of one or more recorder tokens is expressed in a system UI recording language, whereby the system UI recorder
30 program interoperates with the second software program.



Smart UI Recording and Playback Framework

Abstract

Techniques and tools are described for recording and reliably replicating graphical user interface ("GUI") activity for various applications, including applications that have dynamic user interfaces. The techniques and tools may be used in combination or separately. For example, a recording tool receives internal macro data from a GUI-based application as opaque tokens, which are embedded into playback code. A playback tool executes the playback code by passing the embedded tokens back to their original application for playback by the application's internal recorder.



440

140

MOdel 6/30/04 Declaration (2) of MS008158.01

EXPRESS MAIL LABEL NO. EV 331582587 US

DECLARATION AND POWER OF ATTORNEY
FOR PATENT APPLICATION

ATTORNEY DOCKET NO. 3382-67846-01

MS DOCKET NO. 308158.01

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) or 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: **SMART UI RECORDING AND PLAYBACK FRAMEWORK**
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 35, 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 35 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) listed below and those associated with

Customer No. 26119

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

Send Correspondence to:

Customer No. 26119

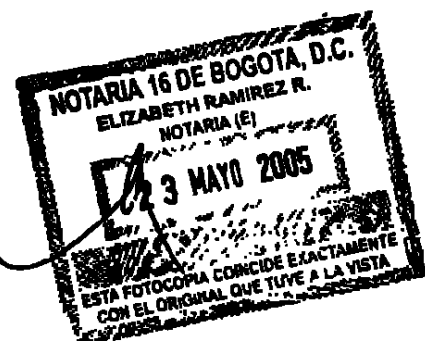
Stephen A. Wight
Kierquist Sperkman, LLP
121 S.W. Salmon Street, Suite 1600
Portland, OR 97204

Direct Telephone Calls To:

Stephen A. Wight
(503) 225-7391

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.

Page 1 of 2



44

141

Model 6/30/04 Declaration (2).doc MS08158.01

EXPRESS MAIL LABEL NO. EV 331582587 US

DECLARATION AND POWER OF ATTORNEY
FOR PATENT APPLICATION

ATTORNEY DOCKET NO. 3382-67840-01

MS DOCKET NO. 308158.01

Full Name of Inventor: Howard E. Herley, III

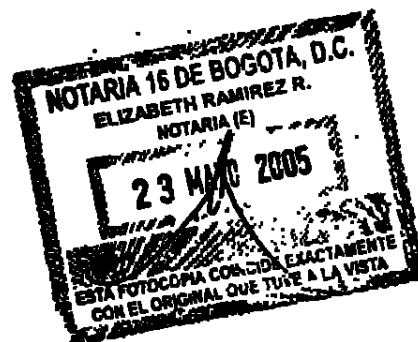
Citizenship: US

Residence: North Bend, WA

Post Office Address: 44415 166th Street SE, North Bend, WA 98945


Inventor's Signature: _____

6/30/2004
Date: _____



142

Figure 1

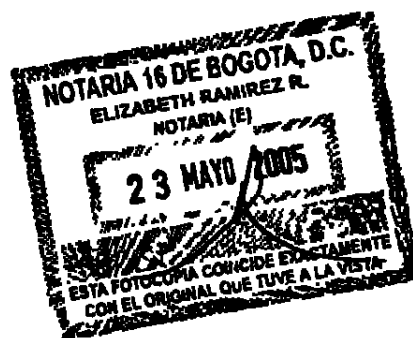
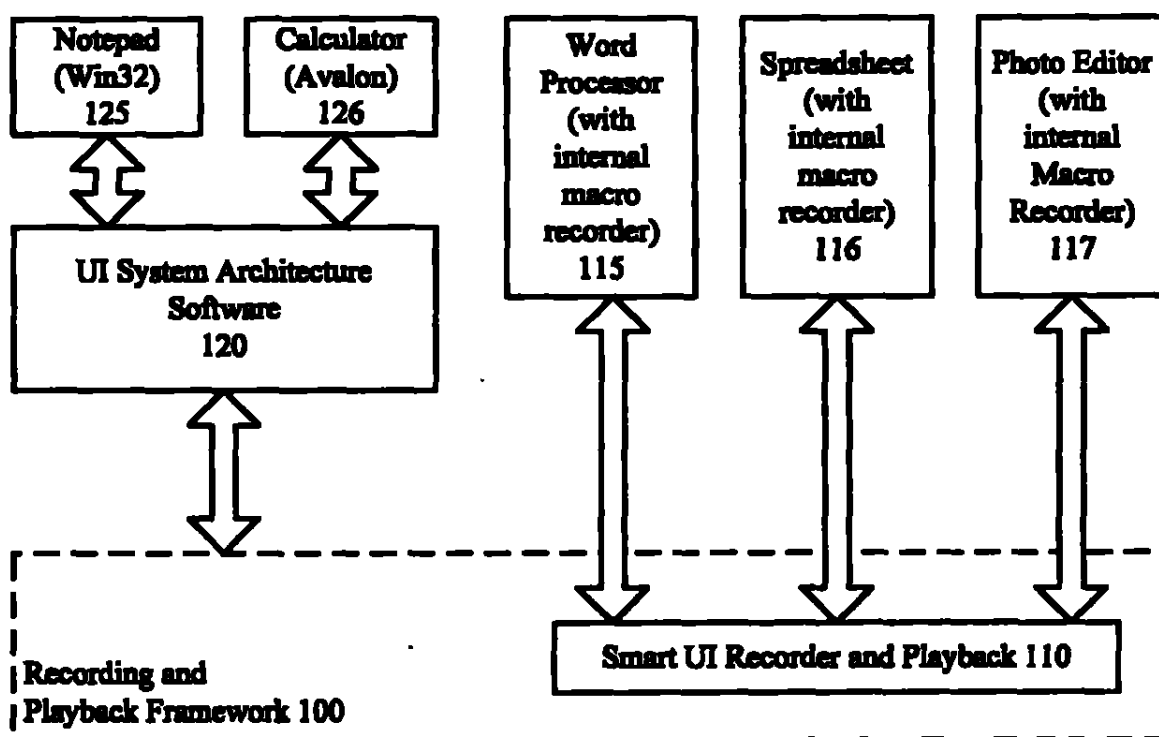


Figure 2

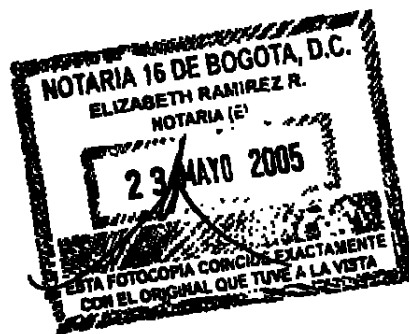
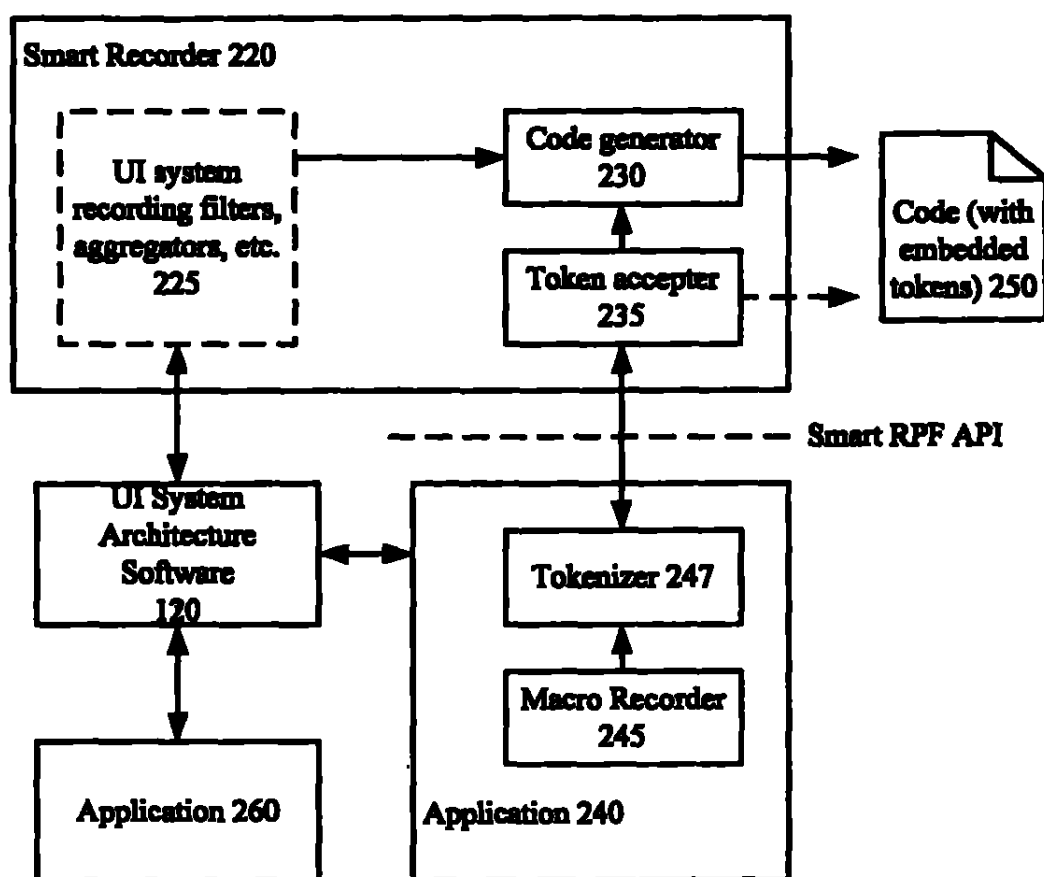


Figure 4

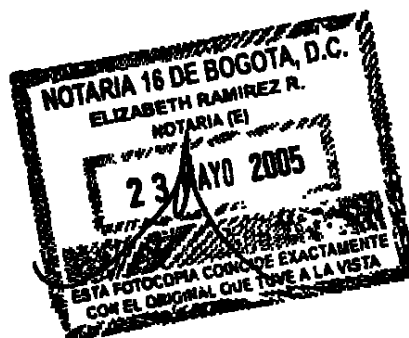
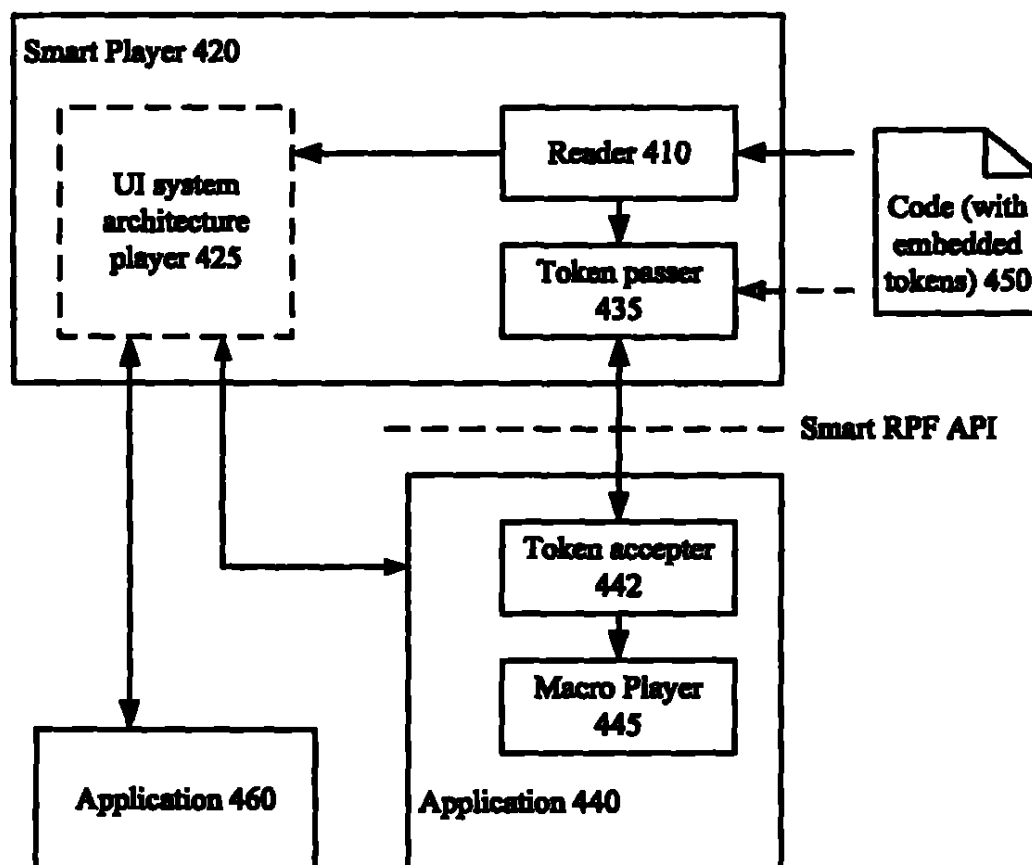


Figure 5

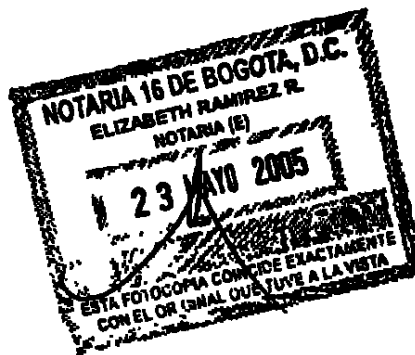
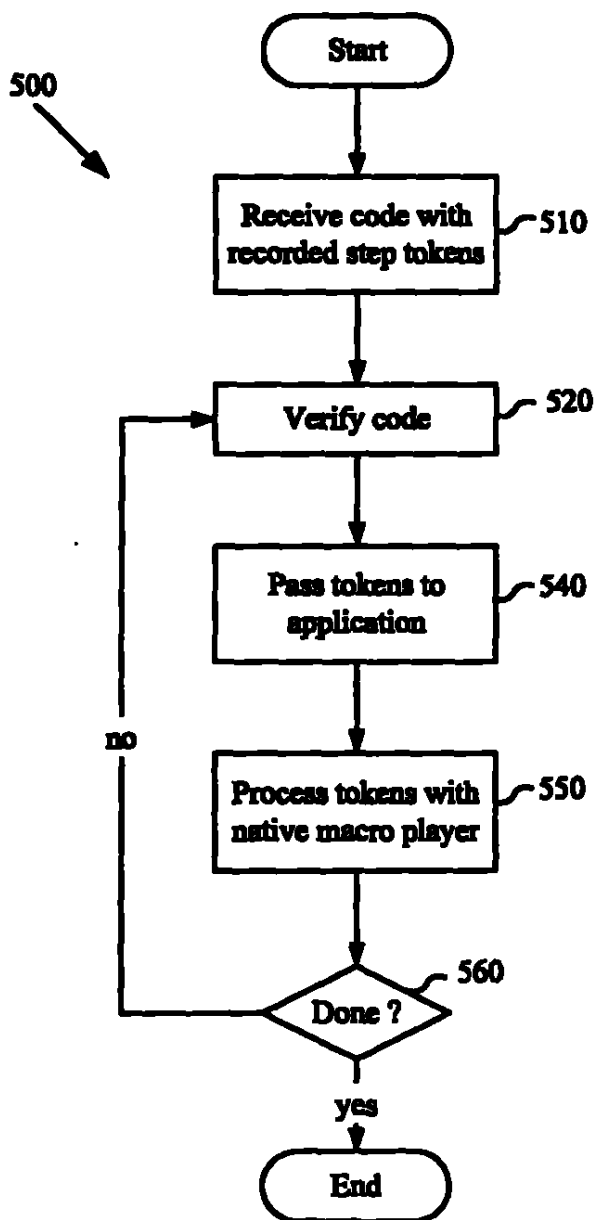
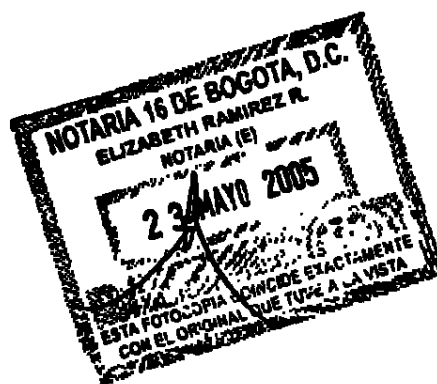
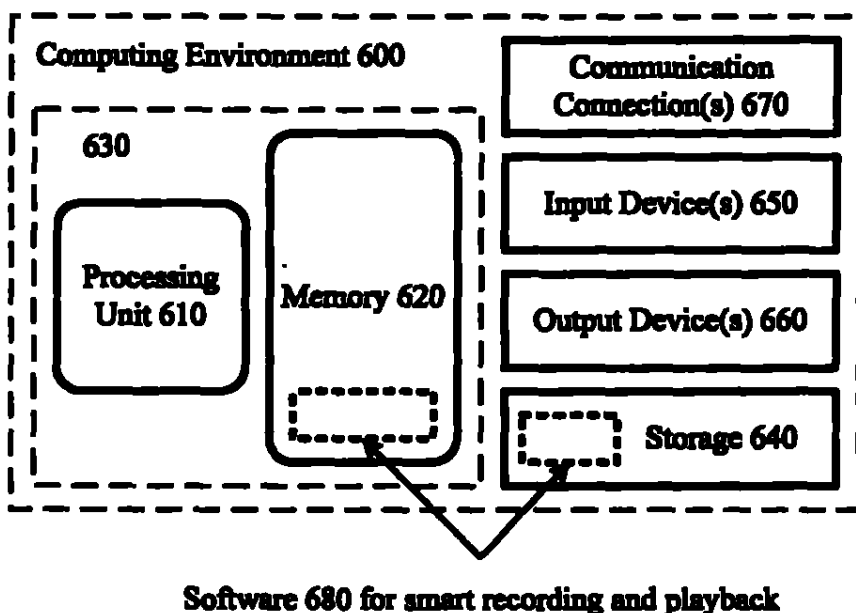


Figure 6



147

Traducción Oficial

LOS ESTADOS UNIDOS DE AMÉRICA

A TODOS AQUELLOS QUE LES PUEDA INTERESAR

**DEPARTAMENTO DE COMERCIO DE LOS ESTADOS UNIDOS
OFICINA DE PATENTES Y MARCAS REGISTRADAS DE LOS ESTADOS UNIDOS**

23 de Marzo de 2005

CON ESTE CERTIFICAMOS QUE EL DOCUMENTO ANEXO ES COPIA VERDADERA TOMADA DE LOS REGISTROS DE LA OFICINA DE PATENTES Y MARCAS REGISTRADAS DE LOS ESTADOS UNIDOS DE LOS DOCUMENTOS DE LA SOLICITUD DE PATENTE ABAJO IDENTIFICADOS, QUE CUMPLIERON LOS REQUISITOS PARA QUE SE LES OTORGARA UNA FECHA DE RADICACIÓN BAJO EL 35 DEL USC 111.

NÚMERO DE SOLICITUD: 10/882,861

FECHA DE RADICACIÓN: 30 de Junio de 2004

Autorizado por el

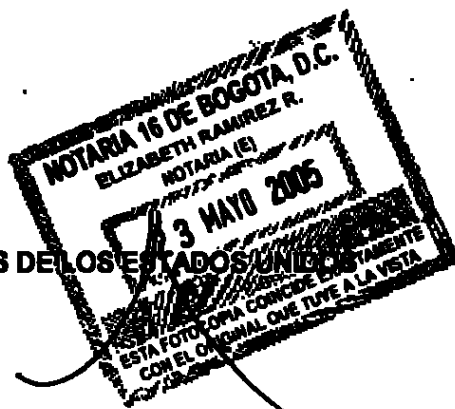
COMISIONADO DE PATENTES Y MARCAS REGISTRADAS

Firma: P. SWAIN

Funcionario Autorizado para Certificar

Sello: DEPARTAMENTO DE COMERCIO

OFICINA DE PATENTES Y MARCAS REGISTRADAS DE LOS ESTADOS UNIDOS



Martha S. Salamanca S.

Martha S. Salamanca S.

Traductora Oficial Juramentada
Resolución No. 835 Minjusticia

147

Traducción Oficial

22713 U.S. PTO
CODIGO DE BARRAS
083004

19587 U.S. PTO
10/882881
CODIGO DE BARRAS
083004

**EN OFICINA DE PATENTES Y MARCAS REGISTRADAS
DE LOS ESTADOS UNIDOS**

COMMISSIONER FOR PATENTS / COMISIONADO PARA PATENTES
P.O. BOX 1480
ALEXANDRIA, VA 22313-1480

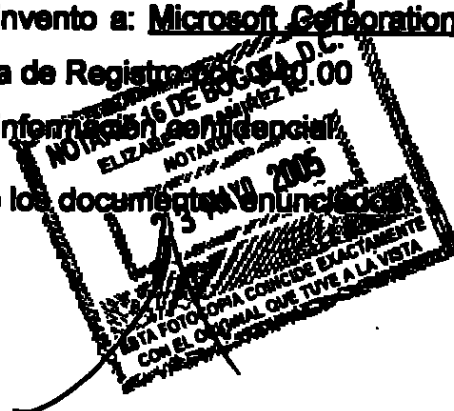
Se transmite para radicación y registro de la solicitud de patente del:

Inventor(s): Howard B. Herdeg, III

Para: Nombre del invento: SMART UI RECORDING AND PLAYBACK
FRAMEWORK/ MARCO DE TRABAJO PARA LA GRABACIÓN Y
REPRODUCCIÓN UI INTELIGENTE

Se Adjunta lo siguiente:

- Hojas de Solicitud y Datos.
- 33 páginas de especificaciones, 5 páginas de Reivindicaciones, un resumen, y una Declaración y poder de Abogado.
- 6 hojas con dibujos.
- Documento de Traspaso del invento a: Microsoft Corporation, hoja de portada para registro y una Tasa de Registro de \$40.00
- Declaración de Publicación de información anticipatoria
- Formato PTO-1449 y copias de los documentos enunciativos



Comunicación - Página 1 de 2

649

Martha I. Samiende S.
Martha I. Samiende S.

Traductora Oficial Juramentada
Resolución No. 835 Minjusticia



TASA DE REGISTRO					
	REVINDICACIONES	NÚMERO	NÚMERO		TARIFA
Para	Registradas	Gratis	Extra	Tarifa	BÁSE
Número Total de Revindicaciones	27	20 =	7	\$ 18.00	\$770.00
Reivindicaciones independientes	4	3 =	1	\$ 88.00	\$ 88.00
Tasa para múltiples reclamaciones				\$290.00	
TOTAL TASA DE REGISTRO					\$982.00

- Se adjunta el Formato PTO-2038 para Pago con Tarjeta de crédito por \$1,022.00 para cubrir la cuota de Registro y Traspaso.
- El Beneficiario de esta solicitud de Traspaso es o será la empresa Microsoft Corporation, y los solicitantes piden que esta información se incluya en la solicitud a publicar.
- Diríjase toda la correspondencia a la dirección asociada con

El Cliente Número 26119

- Por la presente se autoriza al Director para cobrar cualquier tasa adicional que se requiriera para el registro de esta solicitud de Traspaso; además para que consigne cualquier saldo a favor por sobre pago a la Cuenta No. 02-4550. Se adjunta copia de esta hoja.
- Por favor regrese la tarjeta postal adjunta para confirmar que los documentos enunciados fueron recibidos.

Respetuosamente,

KLARQUIST SPARKMAN, LLP

One World Trade Center, Suite 1800
121 S.W. Salmon Street
Portland, Oregon 97204
Teléfono: (503) 228-7391
Fax: (503) 228-8448

cc. Expediente de Cliente No.(308158.01)



Comunicación - Página 2 de 2

680

Martha F. Samiello S.
Martha F. Samiello S.

Traductora Oficial Juramentada
Resolución No. 835 Minjusticia



150

TRADUCCIÓN OFICIAL

Hoja de Solicitud y Datos

Información de la Solicitud de Patente

Solicitud Número:

Fecha de radicación:

Tipo de Solicitud:

Asunto - Tema

Regular

Utilidad

CD-ROM o CD-R

No

Número de discos de CD

Número de copias de CD

¿Formulario Legible por computador (CRF)?:

Nombre:

SMART UI RECORDING AND
PLAYBACK FRAMEWORK / MARCO DE
TRABAJO PARA LA GRABACIÓN Y
REPRODUCCIÓN UI INTELIGENTE
3382-67840-01

Número de Expediente de Abogado:

¿Solicitud de pronta publicación?

No

¿Solicitud de no Publicación?

No

Diseño de Figura Sugerido:

2

Número Total de hojas con Diseños:

6

¿Empresa Pequeña?:

No

¿Se incluye petición?:

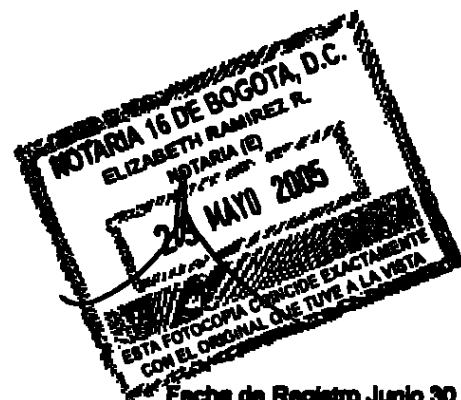
¿Tipo de Petición?:

Agencia Autorizada por el Gobierno de los
Estados Unidos:

Contrato o Números Otorgados:

¿Solicitud de Patente Principal con Orden de
Reserva?:

No



ETIQUETA CORREO EXPRESO No. EV 331582587 US

Fecha de Registro Junio 30, 2004

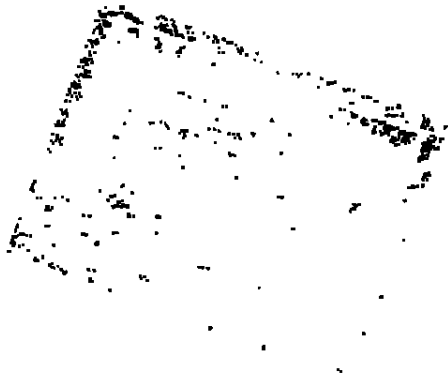
Copia Suministrada por la USPTO de la Base de datos de Imagen IFW el 03/22/2005

131

Martha L. Samiento L.

Martha L. Samiento L.

Traductora Oficial Juramentada
Resolución No. 835 Mijusticia



151

Información de los Solicitantes

Tipo de Especialidad del Solicitante:	Inventor
País de Origen:	Estados Unidos
Estatus:	En plenas facultades
Primer Nombre:	Howard
Segundo Nombre:	B.
Apellido:	Herdeg
Sufijo del Nombre/Sudónimo:	III
Ciudad de Residencia:	North Bend
Estado o Provincia de residencia:	WA
País de Residencia:	Estados Unidos de América
Dirección de Correo:	44415 166 th Street SE
Ciudad:	North Bend
Estado o provincia para correo:	WA
País para envío de correo:	Estados Unidos
Código Postal:	98045

Información para Correspondencia

Número del cliente para correspondencia: 26119

Información del Representante Legal

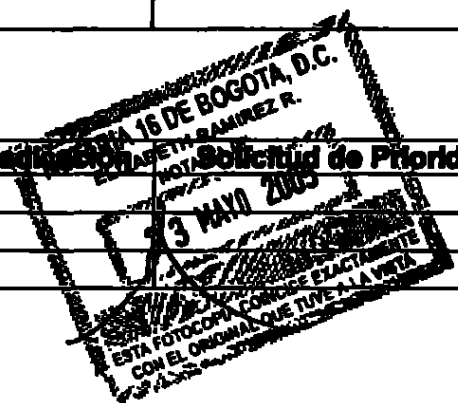
Número del Representante del Cliente: 26119

Información Prioritaria Interna

Solicitud:	Tipo de Continuidad	Solicitud Principal	Fecha de Radicación de la solicitud principal

Información Prioritaria Externa

País:	Solicitud Número:	Fecha de Radicación	Solicitud de Prioridad



ETIQUETA CORREO EXPRESO No. EV 331582587 US

Fecha de Registro Junio 30, 2004

Copia Suministrada por la USPTO de la Base de datos de Imagen IFW el 03/22/2005

152

Martha L. Samiento S.
Martha L. Samiento S.

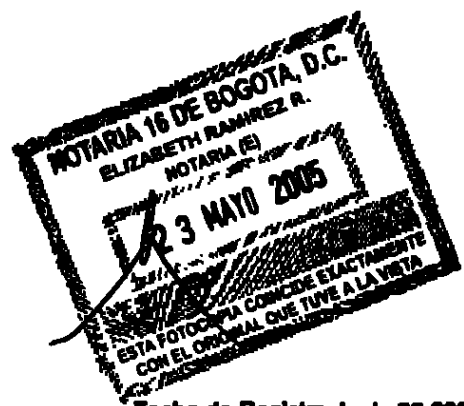
Traductora Oficial Juramentada
Resolución No. 835 Minjusticia



52

Información del Beneficiario de la Patente

Nombre del Cesionario	Microsoft Corporation
Dirección:	One Microsoft Way
Ciudad para envío de correo:	Redmond
Estado o provincia para envío de correo:	WA
País para envío de correo:	Estados Unidos de América
Código Postal:	98052



ETIQUETA CORREO EXPRESO No. EV 331582567 US

Fecha de Registro Junio 30, 2004

Copia Suministrada por la USPTO de la Base de datos de Imagen IFW el 03/22/2005

153

Martha S. Samiando S.

Traductora Oficial Juramentada
Resolución No. 835 Mhjusticia



153

Marco de Trabajo para la Grabación y Reproducción UI Inteligente

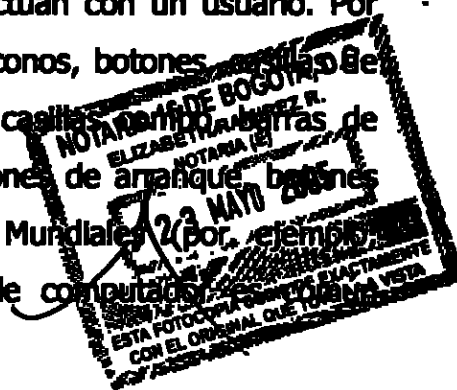
Campo Técnico

Para grabar actividades en varias aplicaciones de interfaces gráficas de usuario se describen varias técnicas y herramientas. Por ejemplo, una herramienta de grabación que abarca todo el sistema sustenta aplicaciones de macros nativas o grabación UI como también grabaciones UI de macros no nativas.

Antecedentes

1. Interfase de Usuario Gráfica

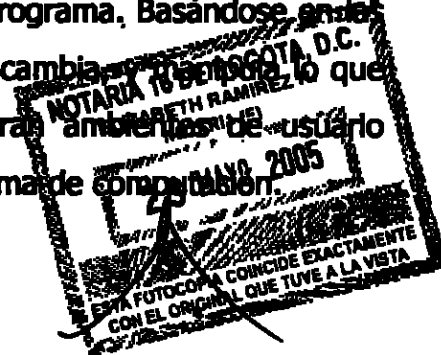
En muchos sistemas de computación modernos, los usuarios interactúan con programas de software a través de una interfase de usuario gráfica ("GUI"). Básicamente, un GUI es una Interfase entre un usuario y un computador que utiliza figuras en vez de palabras para solicitar entradas del usuario y presentar programas de salida. Un GUI típico está compuesto de elementos de Interfase ("elementos UI"), que son esos aspectos de un sistema de computación o programas que se ven, se oyen, se perciben o interactúan con un usuario. Por ejemplo los elementos UI incluyen items tales como, iconos, botones, casillas de diálogo, casillas editables, casillas que se despliegan, casillas con barras de desplazamiento, listas que se pueden seleccionar, botones de arrastre, botones de radio y varios componentes de Páginas Web Mundiales (por ejemplo, hipervínculos e imágenes). En un programa típico de computadores encontrar literalmente miles de elementos UI.



154

Aunque un elemento individual de un GUI puede aparecerle al usuario como un ítem único, puede realmente consistir de un número de ítems separados o subelementos que serán combinado. Por ejemplo, un ítem de la barra herramientas puede consistir de un elemento de lista, un elemento de casilla, un elemento de barra de desplazamiento, etcétera. Aún más, cada uno de estos subelementos pueden estar compuestos de otros subelementos. De esta forma, los elementos UI pueden servir como bloques de construcción para construir otros elementos UI más complejos. Este enfoque es útil porque el software que administra la interfase de usuario puede reutilizar las definiciones de ciertos elementos comunes cuando los ensambla en un elemento compuesto.

Muchos elementos UI en un ambiente GUI representan rasgos de un programa y se muestran en el monitor del computador para que el usuario interactúe con el programa seleccionando, marcando, accediendo y operando con ellos. Esta interacción de usuario se hace manipulando un puntero en la pantalla (controlado típicamente por un ratón o un teclado) y presionando o haciendo clic con el botón mientras el puntero señala un elemento UI. Por ejemplo, en un procesador de palabras, un usuario puede manipular el ratón para seleccionar un ítem en la barra de menús del programa, haciendo clic en un ícono de la barra herramientas, marcando un bloque del texto en la ventana de visualización. En forma similar, un usuario puede utilizar el teclado para interactuar con una aplicación de computador, por ejemplo, en un programa de procesador de palabras, un usuario puede presionar "ALT-F," o "CTRL-B," o cualquier otra combinación de teclas definidas para acceder a distintas funcionalidades del programa. Basándose en las entradas del ratón o el teclado, el computador añade, cambia, o elimina lo que se muestra en la pantalla. Tecnologías GUI suministran ambientes de usuario amigables para que los usuarios interactúen con el sistema de computador.

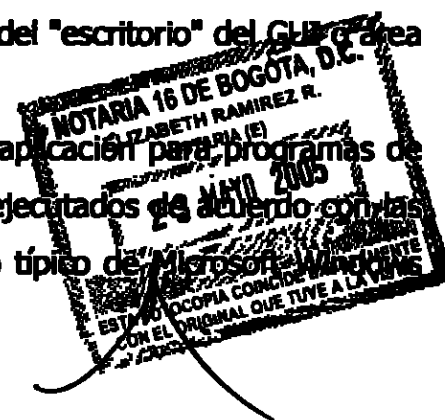


2. Automatización UI

La automatización UI ("UIA") es un marco de trabajo asequible para Microsoft Windows que pretende dirigirse a las necesidades de los productos de tecnología de asistencia y marcos de trabajo de pruebas automatizadas suministrando acceso programático a la Información acerca de la interfase del usuario ("UI"). Por ejemplo AUI le permite a un programa lector de pantalla acceder a la información acerca del UI de un procesador de palabras, suministrando a un programa de lectura la información que necesita para suministrar pistas audibles a un usuario con problemas de visión. A través de una serie de métodos de aplicación de programación de interfases ("API"), del UIA suministra un mecanismo bien estructurado para crear e interactuar con el UI. Los diseñadores de controles de aplicaciones utilizan el conjunto API UIA para hacer que sus productos sean más accesibles a distintos usuarios a través de software nuevo o existente (escritos potencialmente por otras personas) para acceder a los menús del programa y a otros elementos UI. Por ejemplo, pantallas braille, lectores de pantalla (narradores), aumentadores de tamaño, y otros tipos de software pueden utilizar UIA en Microsoft Windows para facilitar al usuario el uso del computador al cual de lo contrario no tendría acceso.

En la práctica, la Automatización UI utiliza una jerarquía de elementos UI localizados en una estructura de árbol para suministrar información UI confiable al sistema operativo de aplicaciones del computador. Los elementos de una GUI se pueden considerar como anidados dentro de ellos mismos para describir su organización en forma precisa. Por ejemplo, en la cima de una estructura de árbol hay un elemento de escritorio que es representativo del "escritorio" del ~~GUI~~ área de fondo por defecto.

El elemento de escritorio tiene varios elementos de aplicación para programas de aplicación que se han Invocado si están listos a ser ejecutados de acuerdo con las instrucciones del usuario (por ejemplo, un escritorio típico de Microsoft Windows).



156

157

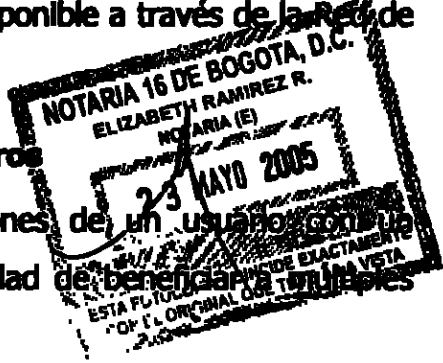
Estas técnicas identifican clases de objetos de elementos UI o tipos, pero no suministran un identificador fuerte que identifique en forma única un elemento UI a través de un reinicio del computador que corre el programa, a través de una construcción diferente del programa cuando todavía está en desarrollo, a través de abrir otra instancia del mismo programa, o para abrir el mismo programa en otro computador.

La UIA se sobrepone a estas deficiencias generando un ID compuesto que identifica en forma única a los elementos UI en el árbol GUI. La UIA genera un identificador compuesto añadiendo Información de Identificación (por ejemplo, un nombre de control o un tipo de control) que está asociado directamente con un elemento UI para una información de identificación jerárquica (por ejemplo control paterno, control de los hijos, y/o control de los hermanos) e información de control con un patrón específico (por ejemplo, la profundidad de un elemento UI en el árbol). Por ejemplo, un identificador para un elemento UI objeto se puede generar recolectando información de identificación que se relaciona con el elemento paterno GUI que describe el arreglo jerárquico entre un elemento de páginas UI y el elemento raíz del escritorio. A través del concepto de una ruta, el identificador relacionado de la jerarquía única del elemento UI y su paternidad se pueden nivelar para identificarlo en forma única y persistente.

El identificador único (ID persistente) suministra un fácil acceso a los elementos individuales UI de tal manera que la funcionalidad del programa que alberga los elementos UI se puede programar y probar, con lo cual el elemento particular UI se puede identificar con otros módulos del programa. Para información adicional acerca de UIA ver, por ejemplo, la documentación disponible a través de la Red de Diseñadores de Microsoft.

III. Grabadoras de GUI y Constructores de Macros

La habilidad de grabar y reproducir las interacciones de un usuario con un computador en un ambiente GUI tiene la potencialidad de beneficiar a múltiples

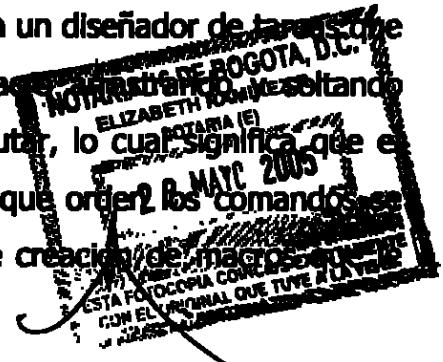


158

usuarios, incluyendo a negocios (u otras organizaciones grandes), usuarios, diseñadores de software, bancos de prueba, y personal de apoyo a los computadores. Por ejemplo, las organizaciones de negocios pueden perfilar sus procesos de negocio, tales como uso de software para la administración de cadenas de distribución, automatizando gran parte del proceso. Los usuarios se benefician creando macros o libretos que combinan una serie de acciones imputadas en una acción de reproducción que se dispara con un solo paso. Los diseñadores de software pueden utilizar la habilidad para grabar las acciones de un usuario para ayudar a generar casos de prueba para software que está en desarrollo. Los probadores pueden grabar y reproducir herramientas para construir pruebas para ejecutar regresiones automatizadas de prueba. Personal de apoyo a los computadores pueden grabar las acciones de un usuario para descubrir la razón por la cual un computador se bloquea, o para ayudar a los usuarios a entender cómo utilizar el software.

A. Diseñadores Comerciales de Macros

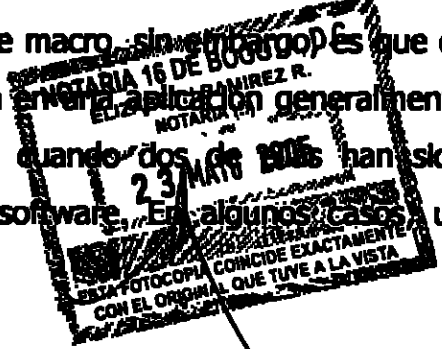
Muchos programas convencionales de diseño de macros generan guiones que muestran los comandos internos y acciones ejecutadas por un computador para ejecutar una función en una aplicación. Sin embargo, en muchos casos, un usuario puede desarrollar guiones en forma independiente basados en una serie de comandos en los guiones y construcciones complejas de programación. Por lo tanto, los usuarios tienen que entender la lógica de programación y, hasta cierto punto, la lógica de los programas que se controlan para crear y utilizar el macro. Por ejemplo, AutoMate, un programa de macros, utiliza un diseñador de macros que utiliza el arrastre-y-suelta para crear guiones de macros, arrastrando y soltando pasos específicos en el orden en que se deben ejecutar, lo cual significa que el usuario de AutoMate tiene que entender cómo y en qué orden los comandos se deben emitir. El Macro Scheduler es un programa de creación de macros que



permite al usuario escribir un macro utilizando más de 200 guiones de comandos y programando construcciones (que no incluyen variables actuales declaradas y otras estructuras definidas por usuario). La complejidad que se requiere para crear y editar un libreto generado por estos programas y el hecho de que los guiones generados por estos constructores de macros no representa una grabación legible de la actividad UI que le quita a esos programas de macros su utilidad, particularmente a usuario nuevos o al personal de apoyo y diseñadores de software que tratan de resolver problemas.

B. Lenguaje Interno Macro

Como una alternativa al software comercial de diseño de macros, muchas aplicaciones tienen la habilidad inherente de grabar y reproducir macros utilizando su propia aplicación especial de lenguaje de control ("lenguaje macro"). Una aplicación de lenguaje macro es típicamente única a la aplicación y generalmente está basada en el modelo de objetos interno ("IOM"). Mientras es posible construir un lenguaje de macro sin técnicas orientadas hacia los objetos, la mayoría de modelos internos utilizan una representación orientada hacia los objetos de la estructura del programa. El IOM provee un perfil o modelo de clases, atributos, operaciones, parámetros, relaciones, y asociaciones de objetos subyacentes al programa. El lenguaje macro accede a sus aplicaciones IOM y se engancha a su mecanismo de comunicación (tales como llamadas de eventos) para acceder y llamar a las distintas funcionalidades de la aplicación. Por ejemplo, un usuario de Microsoft Word puede grabar un macro para formatear texto automáticamente. Uno de los inconvenientes principales del lenguaje macro, sin embargo, es que es específico a la aplicación. Un macro que se graba en una aplicación generalmente no corre en otras aplicaciones, particularmente cuando dos de ellos han sido desarrolladas por compañías que compiten en software. En algunos casos, un



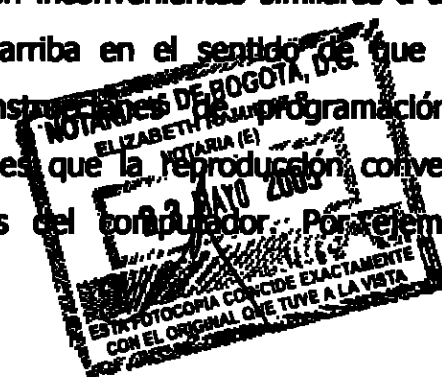
macro que se graba para una versión de una aplicación no encuentra soporte en una versión futura de la misma aplicación.

C. Dependencia en los Macros

Desarrollar macros puede ser difícil, y muchos usuarios y negocios son renuentes a hacer cualquier cosa que pueda dañar los macros existentes. Los negocios en particular son renuentes a hacer algo que pueda afectar en forma negativa sus procesos de negocios. De hecho, muchos usuarios y negocios son renuentes a actualizar el software o incluso a instalar parches por temor a a "dañar" algo. Por ejemplo, supongamos que un negocio emplea un proceso automático de negocios (por ejemplo, en la forma de un macro) que escanea los faxes que recibe para producir archivos de texto de los faxes, compara los datos en el archivo de texto a entradas en una hoja de cálculo para verificar la información de cuentas, y finalmente envía un nivel de confirmación al remitente del faxes. Este proceso de negocios utiliza generalmente un conjunto de macros potencialmente complicados y una variedad de paquetes de software para funcionar en forma apropiada (por ejemplo, un programa de escáner, un OCR, un programa de hoja de cálculo, y un programa de archivos de texto, etc.). Los negocios y los usuarios se sienten incómodos actualizando macros o software al menos de que tengan seguridad de que la inversión actual en automatización de los procesos permanecerá intacta.

D. Otras Grabadoras UI

Muchas grabadoras convencionales UI tienen inconvenientes similares a aquellos de las herramientas de macros descritos arriba en el sentido de que utilizan comandos de guiones complejos y construcciones de programación para representar los datos. Otro inconveniente es que la reproducción convencional depende de las condiciones preexistentes del computador. Por ejemplo, la



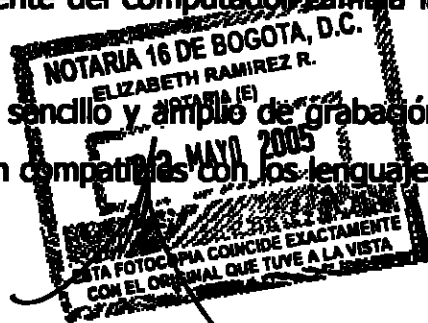
161

reproducción puede depender de cierta configuración de hardware, instalaciones de software, y/o el estado dinámico del ambiente de ejecución (tal como la disponibilidad o localización de elementos UI para una grabación particular de una interacción). Utilizando una herramienta de reproducción convencional, cualquier cambio a esas condiciones preexistentes pueden causar que la reproducción se detenga.

Por ejemplo, supongamos que un usuario reconfigura la interfase de una aplicación. Algunas aplicaciones basadas en GUI le permiten al usuario mover botones, reconfigurar menús, añadir o remover otros elementos UI para distintos rasgos del programa, añadir vínculos a los macros en la barra de menú, o ejecutar otras alteraciones UI. Aunque la habilidad de personalizar menús es un rasgo útil (por ejemplo permitiendo a los usuarios configurar su UI a sus necesidades específicas), puede causar muchas fallas en la grabación y la reproducción porque los elementos UI ya no están en los sitios esperados.

Por ejemplo, el grabador de macros de Microsoft Windows 3.1 intenta replicar las acciones de un usuario en un GUI grabando los movimientos del ratón y las coordenadas del ratón cuando se hace clic en uno de sus botones. En la reproducción, si un elemento GUI no esta donde se espera, la reproducción fallas. Para mayor información en la grabación de macros de Microsoft Windows 3.1 ver, por ejemplo, las referencias denominadas, Guía del Usuario para Trabajos en Grupo de Microsoft Windows, pág. 137. En otro ejemplo, suponga que un usuario graba una actividad GUI en una resolución baja del monitor/escritorio. Más tarde, cuando el usuario cambia a una resolución más alta del monitor/escritorio la reproducción va a fallar porque la posición de los elementos UI en la pantalla ha cambiado. Por lo tanto, en la medida que la ambiente del computador cambia la reproducción se hace cada vez menos confiable.

En conclusión, existe la necesidad para un sistema sencillo y amplio de grabación de macros y herramientas de grabación UI que sean compatibles con los lenguajes



162

macro existentes y grabadoras de macro específicas a una aplicación. Al mismo tiempo, existe la necesidad para un sistema amplio y sencillo de herramientas de grabación de macros y UI que trabajen con interfaces de usuario dinámicas.

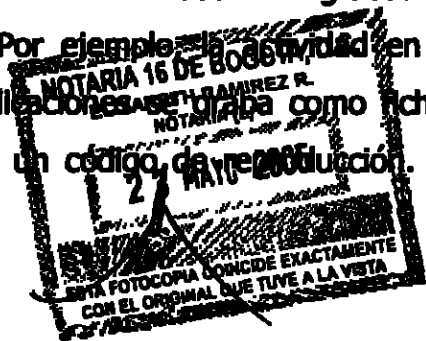
Resumen

Acá se presentan técnicas y herramientas para grabar y reproducir actividades en la interfase gráfica de un usuario ("GUI") para varias aplicaciones en forma confiable. Por ejemplo, una herramienta de grabación de macros y UI simplifica la tarea de construir macros y grabar actividad GUI para algunas aplicaciones mientras apoya las grabadoras de macros específicas a una aplicación , y también apoya la grabación de macros y GUI dentro de una interfase de usuario dinámica.

De acuerdo con un primer aspecto, una herramienta recibe acciones Internas de macro de una interfase gráfica de usuario basada en aplicaciones como fichas opacas. Por ejemplo, un programa de software graba la actividad de un usuario en su ambiente de grabación nativo. Basada en las entradas grabadas, el programa de software genera fichas opacas que se transmiten a la herramienta.

De acuerdo con un segundo aspecto, una herramienta combina fichas grabadas de una acción de macro Interna con otra actividad de la Interfase del usuario. Por ejemplo, un programa de software le pasa acciones de macro internas grabadas como una herramienta de ficha opaca. La herramienta combina esas fichas opacas con fichas recibidas de una grabadora de sistemas UI.

De acuerdo con un tercer aspecto, una herramienta recoge fichas grabadas asociadas con una actividad de la interfase de usuario de distintas aplicaciones y replica la actividad del usuario en la Interfase suministrando cada ficha grabada a la aplicación con la que cada ficha se asocia. Por ejemplo, la actividad en la interfase de un usuario que abarca múltiples aplicaciones se graba como fichas grabadas y se combina con una herramienta en un código de reproducción. La



163

herramienta replica la actividad grabada pasando las fichas grabadas del código de reproducción al software del programa apropiado.

De acuerdo con un cuarto aspecto, una herramienta pasa dos fichas grabadas al programa de software para reproducirlas en la grabadora interna del programa de software. Por ejemplo, unas instrucciones de macro internas se extraen de un conjunto de fichas grabadas y se suministran a la grabadora interna del programa de software asociado con las fichas de reproducción.

Rasgos y ventajas adicionales del invento se harán aparentes de la siguiente descripción de implementaciones que aplican en referencia a los dibujos anexos.

Descripción breve de los Dibujos

La figura 1. es un diagrama de bloque de un marco de trabajo de una interfase de usuario grabando y reproduciendo.

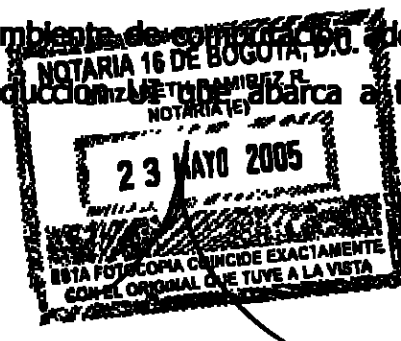
La figura 2. es un diagrama de bloque de una herramienta de grabación UI que abarca todo el sistema.

La figura 3. es un diagrama de flujo que muestra una técnica para grabadora macros nativos a una aplicación con una herramienta de grabación UI que abarca a todo el sistema.

La figura 4. es un diagrama de bloque de una herramienta de reproducción UI que abarca a todo el sistema.

La figura 5. es un diagrama de flujo que muestra una técnica para administrar la reproducción de actividad de macros nativos a una aplicación con una herramienta de reproducción UI que abarca a todo el sistema.

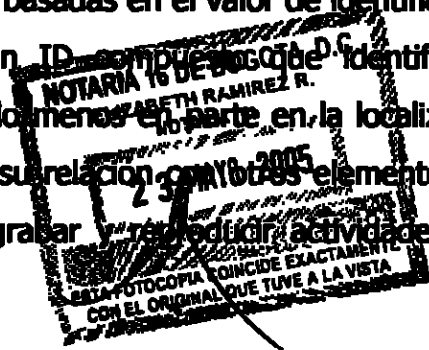
La figura 6. es un diagrama de bloque de un ambiente de configuración adecuado para la implementación de grabación y reproducción UI que abarca a todo el sistema.



Descripción Detallada

La siguiente descripción está dirigida a las técnicas y herramientas para grabación y reproducción de macros y UI para diversos softwares basados en GUI. Por ejemplo una herramienta de grabación de macros y UI que abarque todo el sistema utiliza una técnica de grabación apropiada dependiendo en el tipo de actividad de usuario que se esté grabando y las capacidades de grabación de la aplicación que estén utilizando (a veces referidas acá como grabaciones "Inteligentes"). La herramienta puede utilizar un mecanismo de grabación basado en UIA para grabar la actividad del usuario contra una aplicación compatible con UIA. Si la aplicación no es compatible con UIA (o incluso si es, pero se desea grabar con otros mecanismos), la herramienta puede grabar los pasos simples (por ejemplo, clics con un botón del ratón, entradas del teclado) que el usuario utiliza en su Interacción con una aplicación. La herramienta puede utilizar un mecanismo de grabación especial en varias circunstancias. Estas circunstancias incluyen la grabación de actividades descritas en un lenguaje de macro interno específico a una aplicación (grabado por la aplicación utilizando su propio mecanismo interno de grabación) y grabando las actividades contra una interfase dinámica de usuario. Las herramientas que se describen acá pueden grabar actividad contra varias clases de aplicaciones, software de sistemas operativos, y otro tipo de software.

Solicitud de Patente US Serial No. aa/bb/cc, llamada "herramientas de Grabación/Reproducción para aplicaciones basadas en UI", archivada en junio 15, 2004, cuya divulgación se incorporan acá por referencia, describe varias herramientas de grabación y reproducción. Algunas de estas herramientas utilizan un marco de trabajo UIA para grabar entradas basadas en el valor de identificación de un elemento UI ("UID"). UIA genera un ID compuesto que identifica el elemento UI en un GUI. El UID depende por lo menos en parte en la localización del elemento UI en la interfase del usuario y su relación con otros elementos UI. Cuando la herramienta utiliza un UID para grabar y reproducir actividades que



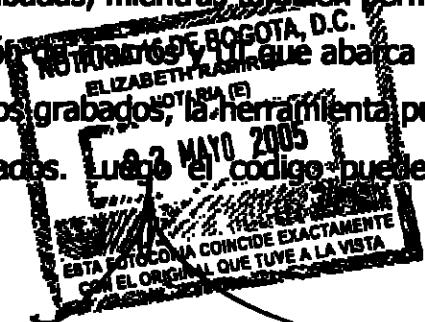
165

165

involucran un elemento específico GUI, cambios en la localización del elemento pueden causar un error en la reproducción.

Por ejemplo, supóngase que un usuario reconfigura la interfase de usuario eliminando un botón en una barra de menú. Cuando una herramienta de reproducción intenta activar el botón que se ha eliminado, una falla en reproducción se origina. Otro ejemplo frecuente es cuando la aplicación se actualiza y ciertos rasgos se renombran para acomodar otros rasgos. Por ejemplo, supóngase que en una versión la función "Abrir Archivo" se mueve para acomodar una función "Archivo Nuevo", y "Abrir Archivo" se convierte en el cuarto ítem en la lista del menú "Archivo". Macros que estén basados en la interfase antigua pueden no funcionar porque el elemento UI se ha movido y/o el UID ha cambiado.

En algunas implementaciones, una herramienta de grabación de macros y UI que abarque todo el sistema utiliza grabación inteligente para resolver este tipo de problemas. Una aplicación graba las acciones de un usuario en lenguaje de macros interno de la aplicación. Luego de grabar, los pasos grabados se expresan en el lenguaje macro interno de la aplicación y se pasan como fichas de pasos grabados de la aplicación a la herramienta de grabación de macros y UI. La herramienta de grabación de macros y UI que abarca todo el sistema puede entonces encapsular las fichas de los pasos grabados y/o asociar las fichas de los pasos grabados con información de marcación. En algunas implementaciones, cada instrucción de macro se convierte en una ficha separada de grabación de pasos. Alternativamente, una serie de instrucciones relacionadas de lenguaje macro se envuelven en una ficha de un paso grabados. Esas fichas mantienen una secuencia temporal de las instrucciones en lenguaje macro subyacente que les permite reproducirse en la aplicación tal como fueron grabadas, mientras también permiten la inter operación con la herramienta de grabación de macros y UI que abarca todo el sistema. Luego de recibir las fichas de los pasos grabados, la herramienta puede generar un código que refleja los pasos grabados. Luego el código puede ser

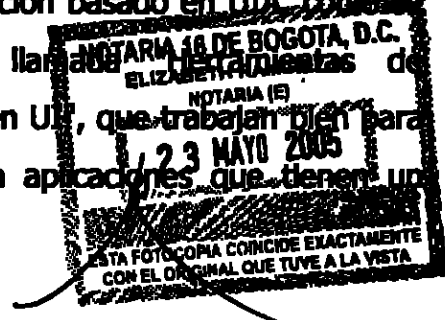


166

reproducido para recrear los pasos grabados o, alternativamente, el código se puede editar para probarse, o corregir fallas u otros propósitos.

Por ejemplo, supongo se que un procesador de palabras tiene unas instrucciones en lenguaje macro "Abrir (nombre de archivo)" que engancha el objeto modelo interno de la aplicación para abrir un archivo con nombre "nombre de archivo". Durante la grabación con la herramienta inteligente de grabación, las instrucciones de macro se pasan a una ficha de grabación de un paso asociada con otra información apropiada, y almacenada para reproducirse, momento en el cual la instrucción subyacente, "Abrir (nombre de archivo)", se pasa a una ficha de un paso grabado de la herramienta inteligente de grabación devuelta a la aplicación. Esto permite a la aplicación utilizar un su propio ambiente nativo de grabación para reproducir y grabar actividades UI dentro de un marco de trabajo de grabación y reproducción UI que abarque todo el sistema. Un macro que abarque todo el sistema se puede escribir de tal manera que no esté sujeto a problemas asociados con cambios en los menús o cambios en la Interfase de usuario o asociados con actualizaciones de software, siempre y cuando el lenguaje macro subyacente permanezca compatible con la versión anterior del software. Aún más, el marco de trabajo para la grabación que abarque todo el sistema provee una interfase de grabación y reproducción unificada. También simplifica la interpolación de macros y grabaciones UI separa distintas aplicaciones en procesos de grabación de macros y UI combinados.

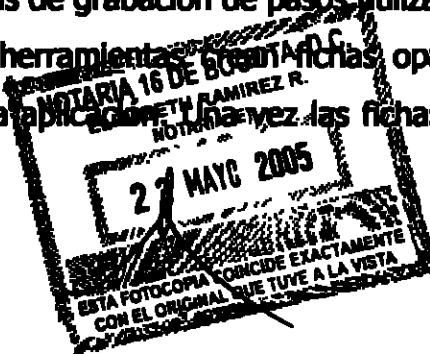
Como parte del marco de trabajo de grabación UI que abarca todo el sistema, cuando una aplicación no utiliza la grabadora de macros interna u otra grabadora de UI específica a una aplicación, se puede utilizar un mecanismo de grabación distinto. Por ejemplo, se utiliza un sistema de grabación basado en UTA, como se describe en la solicitud de Patente de EU llamada "Herramientas de Grabación/Reproducción para Aplicaciones Basadas en UI", que trabajan bien para grabar actividad en la interfase del usuario contra aplicaciones que tienen un



sistema de menús estático. En forma alterna, si una aplicación no apoya las operaciones de acuerdo con la arquitectura de Automatización UI del marco de trabajo RPF para grabar UI a través de todo el sistema puede simplemente grabar los pasos directos ejecutados por un usuario en la medida que el o ella interactúan con la aplicación basada en UI. En algunos casos, varias técnicas de grabación se usan en combinación. Por ejemplo en algunos casos de prueba las entradas directas del usuario, los datos basados en UIA, y los datos de la grabadora interna de macros son todos grabados.

La grabación de pasos directos (por ejemplo, clics del ratón, entradas en el teclado) que un usuario ejecuta interactuando con aplicaciones basadas en UI es relativamente sencilla; el reto está en cómo representar y procesar ésta información cuando una aplicación tiene una interfase de usuario configurable (por ejemplo, una que permite que los menús y los items de menús se cambien a gusto), cuando una aplicación se actualiza y modifica la interfase del usuario, o cuando una aplicación procesa información de macros o grabación de UI en lenguaje interno. Conceptualmente, algunas técnicas y herramientas que se describen acá mejoran la grabación integrando las capacidades internas de grabación de la aplicación con una interfase abstracta y amigable del usuario a través de la cual el usuario puede convertir los pasos grabados y las entradas en un código de reproducción. Como se anotó anteriormente las aplicaciones con sus capacidades internas de grabación pueden invocar su grabadora interna para mapear las acciones de un usuario a los objetos internos de la aplicación, de esta manera resolviendo el "bloqueo de actualización" y problemas de reconfiguración de menús con una simple grabación menú/menú-item.

En algunas implementaciones, macros específicos a una aplicación o datos de grabación UI se recogen y se convierten en fichas de grabación de pasos utilizando una técnica de marcación. Las técnicas y las herramientas para crear fichas opacas definiendo las interacciones del usuario con una aplicación. Una vez las fichas de



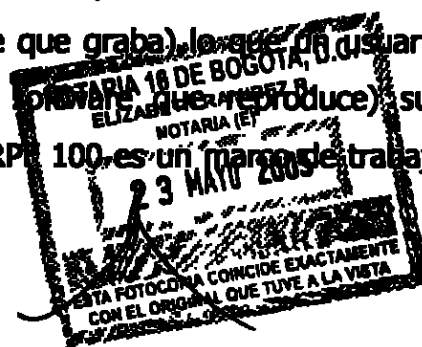
268

los pasos grabados se han generado, otras acciones grabadas (por ejemplo, eventos UIA que se encuentran en fichas) se puede convertir en un código de programación para reproducir en el mismo computador o en otro, potencialmente con fichas de pasos grabados por la aplicación todavía incrustados dentro del código de programación.

En algunas situaciones comunes, las herramientas y técnicas que se describen acá permiten a una herramienta de grabación de macros y UI que abarca todo el sistema ínter operar con varios tipos de programas de software (por ejemplo aplicaciones compatibles con UIA, aplicaciones con grabadoras internas de macros, aplicaciones que no son compatibles con UIA) y varios tipos de interfaces de usuario (por ejemplo, estáticas, dinámicas). Estas técnicas y herramientas hacen la grabación más confiable y robusta mediante el uso de las capacidades internas de grabación de la aplicación cuando son apropiadas. Aún más, las herramientas y técnicas proveen al usuario con una interfase de grabación de macros y UI que es consistente. Las técnicas y herramientas se pueden implementar de varias maneras, y se pueden utilizar combinadas o en forma separada.

I. Marco de Trabajo para la Grabación y Reproducción UI

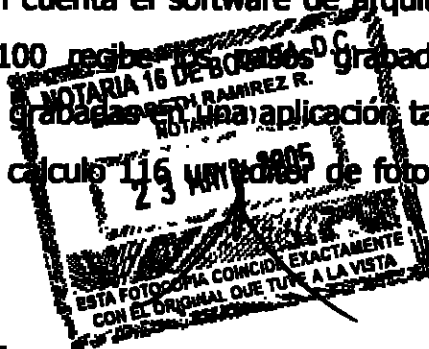
Un marco de trabajo para la grabación y reproducción UI ("UI RPF") es un marco de trabajo extensible designado para facilitar la grabación y reproducción de actividad UI contra aplicaciones u otros programas de software. Módulos que se pueden incorporar dentro del marco de trabajo incluyen filtros, agregados, marcadores, y generadores de código. La figura 1. muestra un UI RPF 100 en donde varias de las técnicas descritas acá se pueden implementar. Básicamente, un UI RPF 100 graba (o trabaja con otro software que graba) lo que el usuario está haciendo y reproduce (o trabaja con otro software que reproduce) sus acciones en el mismo computador o en otro. GUI RPF 100 es un marco de trabajo



(por ejemplo, a nivel de sistema) a través de varias aplicaciones que se puede utilizar con aplicaciones múltiples y en muchas distintas configuraciones, y que está diseñado para facilitar la grabación y reproducción de distintos tipos de actividad del usuario. Es un fundamento en el cual las herramientas de grabación y reproducción se pueden construir. El UI RPF 100 puede utilizar servicios de software del sistema de arquitectura GUI 120, tal como la Automatización UI de Microsoft, para facilitar la grabación UI pero éste no es un requisito del marco de trabajo 100.

La figura 1. muestra varios módulos asociados con el UI RPF 100. El GUI RPF 100 interactúa con el software de arquitectura del sistema UI 120 para convertir los datos UI de entrada grabados tales como clics del ratón, entradas en el teclado, y entradas de las aplicaciones 125, 126 en datos que son más significativos para la grabación y reproducción UI. El marco de trabajo 100 recibe los pasos grabados de aplicaciones tales como un editor de texto 125 una calculadora 126 a través del software de arquitectura del sistema GUI 120, que posiblemente pasa como una ficha que encapsula las acciones de un usuario en la interfase del sistema. En algunas implementaciones el GUI RPF 100 utiliza un mecanismo de grabación y reproducción de GUI descrito en la solicitud de patente U.S. llamada "Herramientas para Grabación/Reproducción de Aplicaciones Basadas en UI", para grabar y reproducir actividad UI recibida a través del software de arquitectura del sistema GUI 120. En forma alternativa, el UI RPF 100 utiliza un mecanismo distinto de grabación y reproducción UI.

Los módulos del UI RPF 100 también incluyen el módulo de grabación y reproducción inteligente 110 para operar con distintas aplicaciones que tienen su grabadora de macros nativa. Sin tener en cuenta el software de arquitectura del sistema UI 120, el marco de trabajo 100 recibe los pasos grabados y otra información de macros en fichas de pasos grabados en una aplicación tal como un procesador de palabras 115, una hoja de cálculo 116, un editor de fotos 117. Las



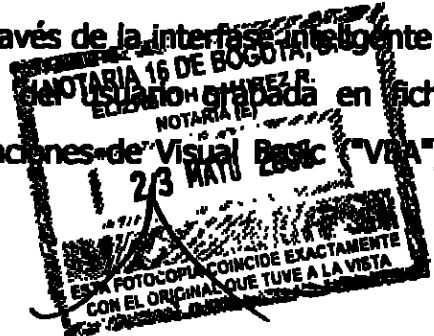
170

170

aplicaciones 115, 116, 117 tienen sus capacidades de grabación internas para mapear las entradas de un usuario a acciones en el ambiente de grabación nativo de la aplicación, cuando las aplicaciones 115, 116, 117 pasan las acciones grabadas internamente como fichas al marco de trabajo 100 a través del módulo inteligente de grabación y reproducción UI 100.

El UI RPF 100 mantiene una lista de las aplicaciones para los cuales actividad de macros o UI se pueden grabar. Una aplicación nueva que es vista por el UI RPF 100 se consulta por lo menos la primera vez que la aplicación se ejecuta, para ver si es compatible con Interacciones a través de una interfase inteligente de grabación. Si la aplicación no es compatible con interacciones a través de la Interfase inteligente, entonces las acciones del usuario se pueden grabar de acuerdo con las capacidades de grabación propias de la aplicación y devueltas a través de la Interfase. En forma alterna, las acciones de un usuario se pueden grabar a través del mecanismo interno de grabación de la aplicación (con datos que pasan a través de la interfase inteligente de grabación) y a través del software de arquitectura del sistema UI 120, donde ambas opciones son posibles. O, Incluso si ambas opciones son posibles, un usuario, una aplicación, o una herramientas que funcionan a través de todo el sistema pueden decidir si utilizan el mecanismo interno de grabación de la aplicación o grabación a través del software de arquitectura del sistema UI 120, pero no ambos. O el usuario, la aplicación, o la herramienta que funciona a través de todo el sistema pueden decidir grabar pasos directos (por ejemplo, movimientos del ratón, clics en los botones del ratón, entradas en el teclado) asociadas con las acciones del usuario, ya sea en combinación o en forma separada con otros datos grabados.

Cuando la actividad de un usuario se graba a través del mecanismo de grabación interno de la aplicación a su vez se graba a través de la interfase inteligente con el UI RPF, la aplicación convierte la actividad del usuario grabada en fichas. El contenido de estas fichas se define en aplicaciones de Visual Basic ("VBA") o en



171

171

otro lenguaje o módulo de objetos utilizado por la aplicación. Últimamente, la Información guardada en esta "carga útil" de las fichas no la interpreta el GUI RPF, pero únicamente las aplicaciones respectivas para la cual la Información se encuentra en su formato nativo. El UI RPF 100 puede convertir en fichas otras fichas de pasos grabados y asociar los datos con otros datos relevantes tales como el nombre de la aplicación.

Con la herramienta que abarca a todo el sistema, los datos de fichas de pasos grabados se pueden concatenar con fichas de pasos grabados obtenidos de otras aplicaciones con una grabadora nativa, o con datos grabados obtenidos a través del software de arquitectura del sistema GUI 120, de tal manera que se pueda implementar la grabación y reproducción a nivel del sistema.

Adicionalmente a grabar macros y datos UI, el marco de trabajo 100 puede producir códigos que reproducen la actividad grabada y/o encapsulan datos que permiten que otro software reproduzca la actividad grabada.

Por ejemplo, en el ambiente de Microsoft Windows, un usuario comienza la grabación de actividad UI, luego hace clic en el botón de "Inicio", y abre un procesador de palabras que trabaja con la interfase Inteligente de grabación. En el contexto de UI RPF, las acciones de un usuario incluyendo los clics del ratón en el botón de "Inicio" se graban. Cuando el procesador de palabras se abre, el UI RPF chequea para ver si el procesador de palabras es compatible con una interfase Inteligente de grabación. Si lo es, el UI RPF invoca la herramienta apropiada dentro del procesador de palabras mismo para grabar las acciones ejecutadas. Si el procesador de palabras puede trabajar con la arquitectura del sistema UI, la herramienta también puede grabar las acciones del usuario (eventos de Automatización UI) en conjunto con el software de arquitectura del sistema UI. Adicionalmente, la herramienta también puede grabar los pasos directos (por ejemplo, clics con el botón del ratón, entradas en el teclado) asociadas por las acciones de un usuario, que pueden ser particularmente relevantes si el

122

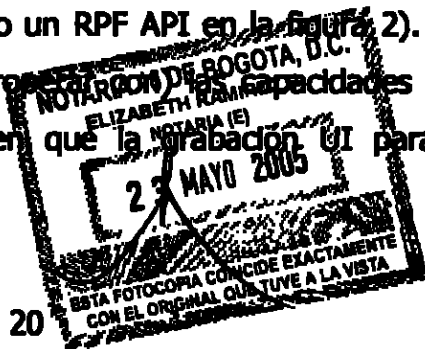
procesador de palabras no trabaja con la interfase Inteligente de grabación y no está habilitado para trabajar con el software de arquitectura del sistema UI. Por lo tanto, el UI RPF puede utilizar mecanismos alternos para grabar reproducir.

Este marco de trabajos 100 puede incluir elementos de software y/o hardware. La relación que se muestra entre los módulos en la figura 1 indica el flujo principal de Información; otras relaciones no se muestran por simplicidad. Dependiendo de la implementación, los módulos se pueden añadir, omitir, romper en múltiples módulos, combinar con otros módulos y/o remplazarse con módulos parecidos. En forma alternativa, el marco de trabajo con módulos diferentes y/o otras configuraciones de módulos ejecutan una o más de las técnicas de grabación y reproducción UI descritas en el presente documento.

II. Grabación Inteligente

La figura 2 muestra una herramienta de grabación UI 220 que abarca todo el sistema en conjunto con las técnicas varias descritas acá con los cuales se puede implementar. El UI RPF de la figura 1 incluye la herramienta de grabación 220. En forma alternativa, la herramienta de grabación 220 se usa como parte de otro marco de trabajo UI.

La herramienta de grabación 220 abarca todo un sistema en el sentido de que graba información de la arquitectura del sistema UI para grabar UI con ciertas aplicaciones, pero también para trabajar con aplicaciones que graban actividad en la interfase del usuario en su propio ambiente de grabación de macros nativo. Estas aplicaciones de grabación nativas intercambian datos de grabación de macros o UI con la herramienta 220 a través de una interfase especial de grabación nativa (que se muestra como un RPF API en la figura 2). Esto permite que la herramienta 220 utilice (o interose) las capacidades de grabación propias de la aplicación, que permiten que la grabación UI para un usuario



20

423

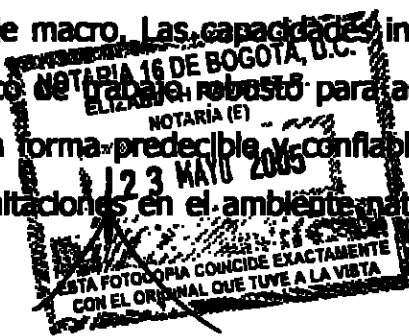
123

dinámico de la interfase y/o grabación UI que se puede reproducir con cambios en la interfase del usuario por cambios de versión.

Esta herramienta de grabación 220 controla el mecanismo que se va usar para grabar macros o datos UI. Por ejemplo, dependiendo de las entradas de un usuario, la configuración del usuario, u otros criterios, la herramienta 220 decide si utiliza la grabación nativa de la aplicación, grabación del sistema UI o ambas para una aplicación particular. La herramienta también puede grabar entradas directas UI del ratón del usuario y del teclado, en forma separada o en combinación con otros mecanismos de grabación. La herramienta 220 también puede grabar datos de macros UI para cada una de múltiples aplicaciones que estén en uso, o para aplicaciones seleccionadas entre aplicaciones múltiples que estén en uso. Por ejemplo, si un usuario configura la herramienta de grabación 220 para que grave únicamente datos de un procesador de palabras, mientras que un usuario graba cortar y pegar de una hoja de cálculo a un procesador de palabras, únicamente los pasos relacionados con el procesador de palabras se graban.

A. Ruta de Grabación Nativa

Para grabaciones nativas de macros o UI, un usuario puede interactuar con la aplicación 240 utilizando dispositivos de entrada tales como un ratón, un teclado, una almohadilla táctil, una pantalla sensible, una pantalla Braille, una palanca de mando, un micrófono, u otros dispositivos similares. En la medida que el usuario interactúa con la aplicación 240, las entradas del usuario se procesan y son grabadas por la grabadora de macros/UI 245 como pasos en el ambiente nativo de grabación de macros de la aplicación, como se expresa en un lenguaje como VBA, PerfectScript, M4, TeX o cualquier otro lenguaje macro. Las capacidades internas de grabación de la aplicación proveen un marco de trabajo robusto para adquirir datos que se puede reproducir la aplicación en forma predecible y confiable. Los pasos grabados sólo están limitados por las limitaciones en el ambiente nativo de



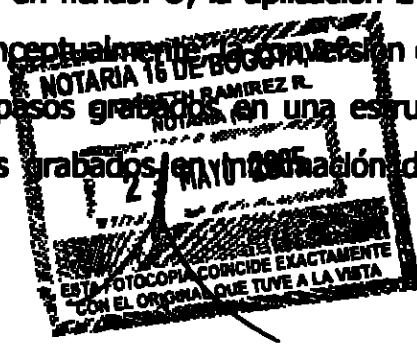
124

174

grabación de la aplicación. Por ejemplo, si una aplicación le permite al usuario declarar variables en un macro, esas variables se pueden grabar y convertir en fichas.

La aplicación 240 convierte en fichas los pasos grabados (que se muestran como la marcación 247) de tal manera que se le pueden pasar a la herramienta 220. Luego, la aplicación pasa las fichas de los pasos grabados a la herramienta 220 a través de la interfase. Las fichas le son opacas a la herramienta 220, lo que significa que la herramienta 220 puede manipular las fichas de los pasos grabados como bloques de datos opacos y discretos cuyo contenido no lo interpreta la herramienta 220. La aplicación 240, por el contrario, puede interpretar y reaccionar al contenido de las fichas de los pasos grabados. El contenido está en el formato nativo de la aplicación 240. Por ejemplo, la aplicación 240 encapsula como una ficha grabada las instrucciones macro para abrir un archivo exactamente como fue grabado (por ejemplo, como datos binarios, comandos de texto, o instrucciones DLL).

La aplicación 240 convierte cada paso grabado y discreto en una ficha, o la aplicación 240 agrupa múltiples pasos grabados en una sola ficha. En cuanto al mecanismo de marcación, la aplicación 240 puede agrupar datos de pasos grabados en uno o más campos de una estructura de datos para la ficha, que luego se pasa como una copia o una referencia a la herramienta 220. La aplicación 240 puede pasar los datos de los pasos grabados como parámetros de un método enunciado, que la herramienta 220 recibe y manipula como datos opacos. La aplicación 240 puede utilizar una hoja de reglas XML (que define una serie de reglas para convertir en fichas los datos de los pasos grabados) para reformatear o reorganizar los datos de los pasos grabados en fichas. O, la aplicación 240 utiliza otro mecanismo de conversión en fichas. Conceptualmente, la conversión en fichas se puede ver como la organización de los pasos grabados en una estructura de datos de fichas, o como envolver los pasos grabados en información de fichas.



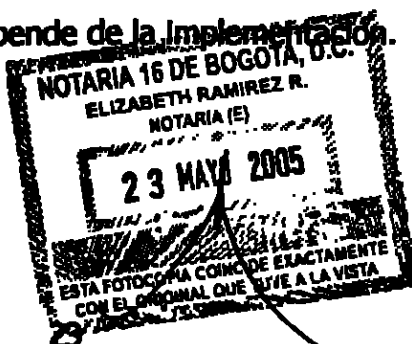
175

195

Esto "esconde" el lenguaje de instrucciones macro de la herramienta, creando una ficha que es independiente del lenguaje macro pero cuyo contenido se puede reconocer en el ambiente de grabación nativo apropiado.

La aplicación 240 puede anexar y añadir datos de marcación a los pasos grabados cuando está convirtiendo los datos en fichas. La marcación de datos provee detalles pertinentes a la reproducción, por ejemplo, información de la aplicación (por ejemplo, el nombre de la aplicación y su versión), una marca de tiempo para indicar el orden, información de privilegios (por ejemplo, que derechos se necesitan para reproducir los pasos), etc.

La herramienta 220 recibe las fichas (que se muestra como el receptor de fichas 235) que vienen de una o más aplicaciones a través de la interfase especial de grabación nativa. Mientras la herramienta 220 no interpreta el contenido de las fichas de los pasos grabados, puede (en vez de o adicional a la aplicación) añadir capas de datos a las fichas de los pasos grabados y/o asociar las fichas con información de marcación tal como el nombre de la aplicación, marcaciones de tiempo, valores de índice, etc. de tal manera que los pasos se pueden reproducir con la aplicación correcta y en la secuencia apropiada. En este sentido, se considera que la herramienta 220 también hacen la función de conversión a fichas. La aplicación 240 es compatible con una interfase nativa de grabación en el sentido de que formatea la información para pasarla a la herramienta 220 a través de la interfase, hace las llamadas apropiadas a través de la interfase, implementa métodos de respuesta para la herramienta 220, etc. La herramienta 220 es compatible con una interfase en el sentido de que implementa métodos para que la aplicación llame, manipule información que viene de otras aplicaciones, inicie métodos de respuesta de la aplicación, etc. El mecanismo exacto para llamar y procesar utilizado por la interfase depende de la implementación.

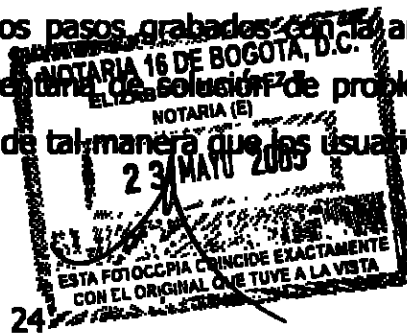


B. Ruta de Grabación del Sistema UI

La aplicación 240 con la grabadora de macros nativa y/o otra aplicación 260 pueden ser compatibles con las interacciones con la herramienta de grabación del sistema 220 a través del software de arquitectura del sistema UI 120. Para grabación con la arquitectura del sistema UI, varios componentes 225 de la herramienta de grabación 220 recolectan (a) datos de entrada del usuario, (b) actividad UI en el monitor (como cambios de foco, cambios de selección, creación de ventanas etc.), (c) información de pistas acerca del ambiente, como aplicaciones que están corriendo, versiones de OS, información de localización, lenguaje UI por defecto, lenguaje UI actual, dispositivos de hardware y configuraciones, modalidad de seguridad, y/o otro tipo de datos que pueden ayudar a describir y recrear el ambiente apropiado para la reproducción, (d) resolver los datos de entrada del usuario en elementos en la interfase del usuario afectados por la entrada del usuario, y (e) expresar la actividad del usuario como eventos lógicos grabados. Estos eventos lógicos grabados ayudan a proveer salidas de UI grabadas y legibles y reproducciones de grabaciones UI confiables. La grabación lógica de eventos se puede considerar como un tipo de fichas de grabación UI, pero a diferencia de las fichas de los pasos grabados opacos con datos de grabación nativos, la herramienta 220 interpreta la grabación lógica de dichos eventos para visualizar, editar, convertir un código, etc.

En algunas implementaciones, la herramienta 220 ejecuta una grabación con la arquitectura del sistema UI con filtros y agregados como se describe en la Solicitud de Patente de E. U. llamada "Herramientas de Grabación/Reproducción para Aplicaciones basadas en UI". De otra manera, la herramienta 220 utiliza otra técnica para la grabación con la arquitectura del sistema UI.

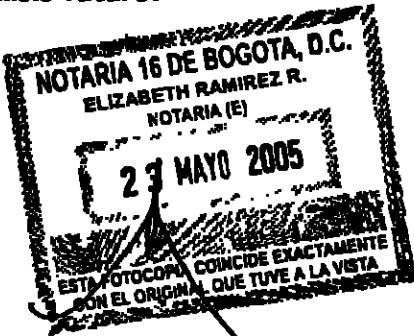
La herramienta 220 puede desplegar los pasos grabados con la arquitectura del sistema UI en un visor tal como una ventana de solución de problemas, un visor de códigos fuente, o un editor de texto de tal manera que los usuarios pueden leer



y buscar errores en los datos. En forma típica, la herramienta 220 no muestra información acerca de las fichas de los pasos grabados con datos de grabación nativos, en la medida que el contenido de estas fichas es opaco a la herramienta 220. Cuando las fichas de los pasos grabados se asocian con información de marcación tal como el nombre de la aplicación, marcaciones de tiempo, etiquetas de acciones u otras anotaciones hechas por el usuario o la aplicación, la herramienta 220 puede mostrar dicha información. Los datos se pueden almacenar en un archivo de registro para análisis futuro.

C. Otras Entradas de Usuario

Los componentes varios 225 de la herramienta de grabación 220 pueden también recoger directamente datos de entrada del usuario y monitorear en forma similar la actividad UI, seguir información acerca del ambiente, tal como aplicaciones que estén corriendo, etc. grabar estos eventos ayuda en situaciones donde una aplicación no es compatible con la interfase inteligente de grabación o la arquitectura del sistema UI. Estas acciones grabadas también se pueden mostrar al usuario en un visor de tal manera que pueda corregir errores en los datos. En forma típica, los datos de las acciones directas del usuario son menos legibles y confiables para reproducción que otro tipo de fichas grabadas como se explicó anteriormente en los antecedentes. Sin embargo, cuando las otras técnicas de grabación son inaplicables, esto provee un mecanismo para grabar datos de aplicación no nativos y no UIA en el UI RPF que abarca todo el sistema. Los datos que se graban de las acciones directas del usuario se pueden almacenar en forma similar en un archivo del registro para análisis futuro.



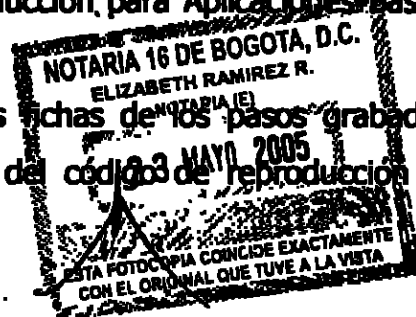
738

D. Código de Reproducción

En algunas implementaciones, la herramienta 220 combina fichas de diferentes aplicaciones y modalidades de grabación en un único archivo mediante el intercalado de datos. Por ejemplo, fichas de pasos grabados para acciones grabadas en forma nativa para distintas grabadoras nativas se combinan en un solo flujo de entrada, junto con las fichas para las acciones en la arquitectura del sistema UI y otras entradas del usuario. En forma típica, las entradas de un usuario se reciben en forma serial y la secuencia de fichas mantiene de esta manera el orden en el flujo. Marcaciones de tiempo, valores de índice, o información de la aplicación, puede marcar las fichas para ayudar a mantener el orden apropiado. O, la herramienta 220 crea archivos separados para distintas aplicaciones y modalidades de grabación.

Para fichas de acciones del sistema UI, la herramienta 220 utiliza un generador de códigos 230 para generar un código de reproducción 250 a partir de las fichas. Esto se puede hacer automáticamente como cuando el usuario selecciona la opción de "Generar Código" en la herramienta 220. El código de reproducción 250 se puede generar en una variedad de formatos, incluyendo C#, XML, C++, JAVA, o cualquier otro lenguaje de programación, macros, guiones, u otros comandos o instrucciones para una interfase de reproducción u otros mecanismos para manejar aplicaciones basadas en UI. Por ejemplo, el generador de código 230 mapea las fichas de acciones del sistema UI con instrucciones correspondientes en el código de reproducción 250. Para más detalles sobre la generación de códigos en un ambiente de arquitectura de sistema UI, ver la solicitud de patente de E.U. llamada, "Herramientas de Grabación/Reproducción para Aplicaciones Basadas en UI", entregada el 15 de junio de 2004.

La herramienta 220 opcionalmente pone las fichas de los pasos grabados para acciones grabadas en forma nativa dentro del código de reproducción 250 tal



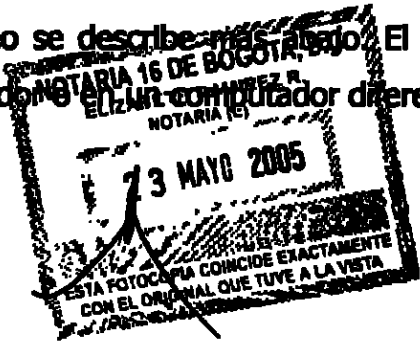
739

139

como un libreto de archivo que mantiene las fichas de los pasos grabados en la secuencia temporal correcta. O, la herramienta 220 utiliza un generador de código 230 para generar el código de reproducción 250 que incluye las fichas de los pasos grabados. Es de anotar, que las fichas de los pasos grabados que encapsulan datos de grabación nativos son opacas a la herramienta 220, por lo tanto estas fichas pasan en forma típica a través del código de reproducción 250, y no son interpretadas por la herramienta 220 para conversión a otras expresiones.

El código de reproducción 250 se puede visualizar, guardar, editar, modificar, añadir a, eliminar de, compilar, o utilizar antes de la reproducción, con las limitaciones esperadas en la medida en que la herramienta puede visualizar, modificar, etc. fichas opacas de pasos grabados. Poder editar el código de reproducción 250 le da a los usuarios un mecanismo conveniente y poderoso para crear macros, generar casos de prueba automáticos, automatizar pasos en un programa, etc. el código generado se puede editar para añadir expresiones condicionales, para cambiar variables, para añadir regresiones, y otras instrucciones de programación al código. Por ejemplo, cuando se va a reproducir, suponga que una pieza del código de reproducción altera un archivo de sistema y requiere al usuario para que se conecte con el Administrador de derechos. De ésta forma, una expresión condicional se añade al código de reproducción, que chequea los derechos apropiados antes de correr una parte del código. En otro ejemplo, supóngase que diseñadores de software quieren probar un programa repitiendo la misma serie de instrucciones una y otra vez, de tal manera, que "una orden circular" se añade al código de reproducción, que hace que el código de reproducción se repita cierto número de veces.

El código de reproducción 250 se puede entonces utilizar para administrar la reproducción de la actividad grabada tal como se describe más abajo. El código 250 se puede reproducir en el mismo computador o en un computador diferente.



130

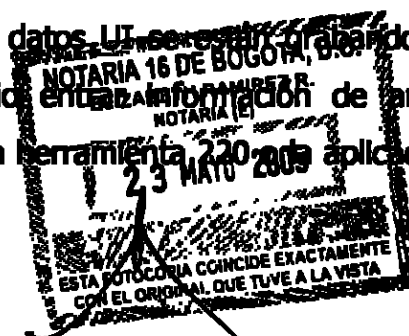
189

E. La Herramienta de Arquitectura y su Uso

En general, los módulos varios de la herramienta 220 se pueden utilizar como un plug-in, los que significa que se pueden remplazar con otros módulos que tengan una funcionalidad similar para distintas aplicaciones. Por ejemplo, el generador de código 230 en la herramienta 220 se puede utilizar como un plug-in, de tal manera que un diseñador puede rehusar partes de la herramienta 220 y cambiar la forma en que el código se genera cambiando el generador de códigos. La herramienta 220 por lo tanto es extensible; su funcionalidad se puede cambiar para distintas aplicaciones. En algunas implementaciones, la herramienta 220 presenta un menú de opciones para módulos plug-in, permitiéndole al usuario controlar la carga de módulos apropiados para la funcionalidad deseada por usuario.

Por ejemplo, en modalidad de grabación "pesada", la herramienta 220 recoge información detallada acerca de las actividades UI y registra la información en forma comprensiva. Esto permite una reproducción más confiable pero es posible que perturbe la ejecución del software en el cual se graba la actividad. O, en una modalidad de grabación "ligera", la herramienta 220 recoge menos información y, por lo tanto, es menos probable que perturbe la ejecución del software donde se graba la actividad. Sin embargo, esto puede dar suficiente información para la reproducción en el mismo ambiente, para mostrar la actividad del usuario y apoyar al personal, o para otros propósitos.

En forma similar, la herramienta 220 puede grabar información en una modalidad interactiva en donde los elementos UI se seleccionan con un rectángulo que los cobija o mostrando en forma intermitente lo que se ha grabado. Otro tipo de retroalimentación que se muestra en forma intermitente se puede utilizar para mostrar al usuario la aplicación para la cual datos UI se están grabando. O, en modalidad interactiva, se le pide al usuario que proporcione información de anotación descriptiva para cada acción o acciones que la herramienta 220 graba en la aplicación 240



189

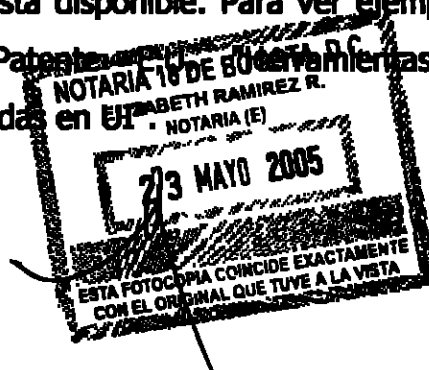
181

no puedan resolver. De otra manera, la herramienta 220 puede operar en el trasfondo para grabar actividad UI sin perturbar al usuario.

La herramienta 220 es responsable de cargar los otros módulos plug-in en la medida que se vayan necesitando (por ejemplo módulos que recogen datos, que convierten datos en fichas, que generan códigos, etc.), imponiendo seguridad (por ejemplo, chequeando los privilegios), desovando trayectorias grabadas, sincronizando actividades entre los módulos, construyendo y manteniendo consultas que se utilizan en la grabación, registrando errores (por ejemplo, a un archivo local u opcionalmente a una red de servicios o un servidor), y recogiendo información acerca de las grabaciones (por ejemplo, versión OS, localización de información, configuración de hardware, etc.).

La relación que se muestra entre los módulos en la figura 2 indica el flujo principal de información; las otras relaciones no se muestran por simplicidad. Dependiendo de la Implementación, los módulos se pueden añadir, omitir, dividir en módulos múltiples, combinar con otros módulos, y/o reemplazar con otros módulos parecidos. De otra manera, una herramienta con distintos módulos y/o otras configuración desde módulos ejecutan una o más de las técnicas de grabación descritas acá.

La interfase del usuario de la herramienta 220 puede ser una interfase a la manera VCR que estaba dispuesta alrededor de la ventana de grabación, anclada a la parte superior o inferior de la pantalla del computador, o minimizada en la barra de tareas. La interfase incluye botones para actividades tales como "Grabar", "Pausa", "Detener", "Reproducir", como también un área para presentar descripciones de los pasos grabados cuando tal descripción está disponible. Para ver ejemplos de tales interfases, de la solicitud de Patente de Estados Unidos No. 2003/018711 A1, titulada "Herramientas para Grabación/Reproducción en aplicaciones basadas en UI".



182

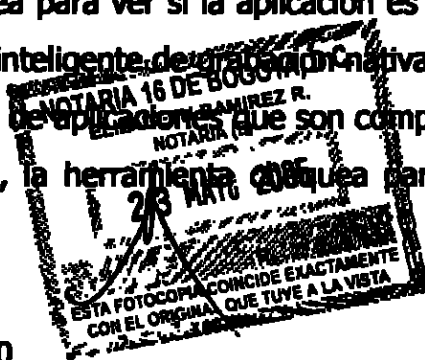
F. Técnicas de Grabación, Cronometraje de las Grabaciones

Existen dos modalidades para transferir las fichas a la herramienta 220 durante la grabación: modalidad en lotes y modalidad interactiva. En general, en la modalidad en lotes, las fichas de pasos grabadas se recogen en la aplicación y se entregan a la herramienta 220 cuando la solicita. Esta solicitud de recoger las fichas puede ocurrir cuando el foco ya no está en la aplicación que graba, y también puede ocurrir periódicamente para distribuir la carga de grabación.

En modalidad interactiva, en contraste, las fichas se envían de la aplicación que se graba a la herramienta 220 tan pronto ocurren. Una desventaja de enviar las fichas en forma interactiva es que aumenta la carga de procesamiento a través en la CPU. En una situación particular, ya sea una de las modalidades interactiva o en lotes es preferible a la otra, dependiendo en la aplicación que se estaba grabando, la cantidad de datos grabados, y que tan rápido se están generando las fichas de los pasos grabados.

La figura 3 muestra una técnica 300 para grabar actividad del macro nativa a una aplicación con una herramienta de grabación UI que abarca todo el sistema. Una herramienta y una aplicación tal como la herramienta de grabación 220 y la aplicación 240 que se muestran en la figura 2 ejecutan la técnica 300. De otra manera, otra herramienta y/o aplicación ejecuta la técnica 300.

La herramienta chequea 310 para ver si la aplicación utiliza una interfase inteligente de grabación nativa. Algunas aplicaciones no son compatibles con las interacciones a través de una interfase. Otras aplicaciones pueden ser compatibles con las interacciones a través de una interfase, pero el usuario o la herramienta determinan no utilizar la grabación nativa. Por ejemplo, cuando un usuario primero inicia una aplicación la herramienta se crea para ver si la aplicación es compatible con interacciones a través de la interfase inteligente de grabación nativa, y si lo es, añade el nombre de la aplicación a la lista de aplicaciones que son compatibles con interacciones a través de la interfase. O, la herramienta chequea para ver si la



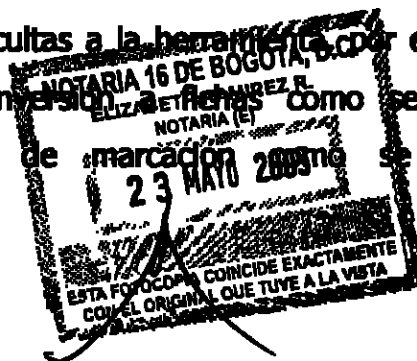
aplicación es compatible con la interfase inteligente de grabación nativa en el momento en el cual se inicia la grabación. Si una aplicación no es compatible con la interacción a través de la interfase, la herramienta puede añadir la aplicación a la lista de aplicaciones que no son compatibles.

Si la interfase de grabación nativa inteligente no se utiliza, la herramienta utiliza otra grabadora 315 para grabar actividad UI. Por ejemplo, la herramienta utiliza una grabadora de software UIA para aplicaciones capaces de UIA, o de otra manera, una grabadora de entradas UI directas para aplicaciones no capaces de UIA. Mientras que la figura 3 muestra las rutas alternativas de grabación, la herramienta puede utilizar ambas, la grabación nativa y la otra grabación simultáneamente.

Cuando la interfase de grabación inteligente nativa se utiliza, la aplicación recibe una entrada del usuario y graba datos macro 320. De esta manera, la grabadora nativa de la aplicación graba los pasos ejecutados por el usuario contra la aplicación.

El proceso de grabar datos macro continúa hasta que la aplicación, con la herramienta, determina 325 que no debe continuar. Este chequeo puede evaluar una variedad de condiciones incluyendo si el usuario ha terminado en forma selectiva el proceso de grabación, si la herramienta ha solicitado fichas de la aplicación en modalidad por lotes, si un contador o timer ha vencido en la modalidad interactiva o por lotes, si hay suministros para convertir en fichas, si un elemento UI designado se ha contactado, si se ha alcanzado un número específico de pasos grabados, etc.

La aplicación luego convierte en fichas 330 los datos de macro grabados, produciendo una o mas fichas que son ocultas a la herramienta. Por ejemplo, la aplicación utiliza un mecanismo de conversión a fichas como se describió anteriormente y/o añade información de marcación 288 no se describió



101

anteriormente. De otra manera, la herramienta ejecuta algunas o todas las conversiones a fichas o adiciones de Información.

La herramienta o la aplicación evalúan 335 si continúan con la grabación de datos de macro. Una vez más, este chequeo puede evaluar una de varias condiciones, incluyendo si el usuario ha terminado en forma selectiva el proceso de grabación, si la conversión a fichas ha sido exitosa o si se debe solicitar Información de anotación del usuario, etc.

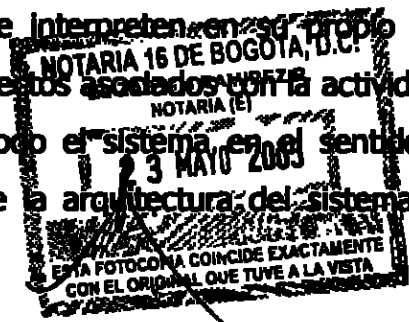
La herramienta opcionalmente convierte 340 las fichas que recibe a un código tal como un guión en el cual las fichas están Incrustadas. Por ejemplo, la herramienta utiliza un mecanismo de generación de código como se describió anteriormente.

De otra manera, las etapas varias de la técnica 300 se ejecutan en forma separada o combinada con otras etapas. Por ejemplo, mientras la figura 3 muestra la conversión a fichas 330 y 340 que ocurre luego de la grabación de datos de macro 320, en forma alternativa una o ambas de estas operaciones se ejecutan en forma concurrente con la grabación 320.

III. Reproducción Inteligente

La figura 4 muestra una herramienta de reproducción 420 UI que abarca todo el sistema en conjunto con otras técnicas varias descritas acá que se pueden implementar. El UI RPF de la figura 1 incluye la herramienta de reproducción 420. En forma alternativa, la herramienta de reproducción 420 se utiliza como parte de otro marco de trabajo UI.

La herramienta de reproducción reproduce la actividad UI recibiendo códigos de reproducción y conduciendo a la aplicación a través del sistema de servicios UI, pasando fichas a la aplicación para que se interpreten en su propio ambiente nativo, o a través de reproducir los pasos directos asociados con la actividad UI. La herramienta de reproducción 420 abarca todo el sistema, en el sentido de que manejan la Información de reproducción de la arquitectura del sistema UI para



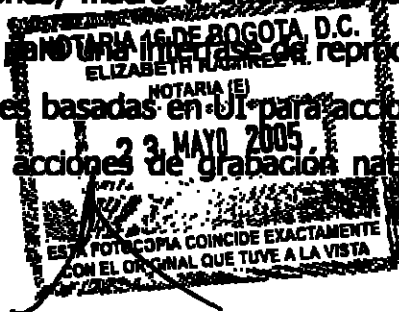
105

reproducción de grabaciones UI con ciertas aplicaciones, pero también inter opera con aplicaciones que reproducen la actividad en la interfase del usuario en su propio ambiente de grabación de macros nativo. Estas aplicaciones de grabación nativas intercambian datos de grabación de macros o UI con la herramienta 420 a través de una interfase especial de reproducción nativa (que se muestra como el RPF API Inteligente en la figura 4). Esto permite que la herramienta 420 controle (y opere con) las capacidades de grabación propias de la aplicación, que pueden permitir la reproducción UI para interfases dinámicas de usuario y/o reproducción UI para interfases que han cambiado con distintas versiones de una aplicación.

La herramienta de reproducción 420 controla las rutas que se han de utilizar cuando se reproducen macros o datos UI. Por ejemplo, dependiendo de que datos se han grabado, la herramienta 420 decide si debe utilizar la reproducción de la grabadora nativa de la aplicación, reproducción en el sistema UI, o ambas. De otra manera, la herramienta también puede reproducir los pasos grabados directamente. La herramienta 420 puede reproducir datos de macros o UI simultáneamente para cada aplicación que está en uso, o para cada aplicación seleccionada entre múltiples aplicaciones en uso.

A. Lector

La herramienta de reproducción 420 recibe el código de reproducción 450 como una entrada de un generador de códigos tal como el generador de códigos 230 que se muestra en la figura 2 y en otras partes. El código de reproducción 450 puede estar en una variedad de formatos, tales como C#, XML, C++, JAVA, o cualquier otro lenguaje de programación, guiones, macro u orientado hacia los documentos, u otros comandos o instrucciones, ~~pero una interfase de reproducción~~ u otros mecanismos para administrar aplicaciones basadas en UI para acciones en el sistema UI. Fichas de pasos grabados para acciones de grabación nativas se



pueden incluir en el código 450 tal como un archivo de guión (que se puede dirigir directamente a la ficha que pasa 435) o Incrustarse en otro código 450.

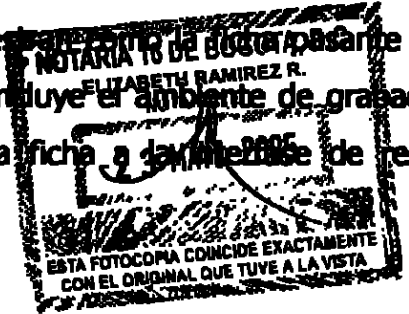
En algunas implementaciones, la herramienta de reproducción 420 recibe un único archivo que combina códigos y fichas de distintas aplicaciones y modalidades de grabación intercalando los datos. Por ejemplo, fichas de pasos grabados para acciones de grabación nativas para distintas grabadoras nativas se combinan en un solo flujo, junto con un código para acciones en la arquitectura del sistema UI y otras entradas de usuario grabadas. O, la herramienta 420 recibe archivos separados para distintas aplicaciones y modalidades de grabación.

En la herramienta de grabación 420, el lector 410 recibe el código de reproducción 450 y verifica que el código 450 esté correcto. Por ejemplo, la herramienta de reproducción recibe un archivo que contiene un código de reproducción e intenta verificar que el archivo esté internamente consistente y consistente con el status actual del sistema. Si se encuentran algunas inconsistencias, se notifica al usuario, y, en algunas implementaciones, se le pide al usuario que complete manualmente los pasos. De otra manera, la reproducción se termina.

Para reproducción de acuerdo con la ruta de la arquitectura del sistema UI, el lector a 410 le pasa el código 450 al reproductor de la arquitectura del sistema UI 425 para control programático de la aplicación como se especifica en el código 450. Para reproducción de acuerdo con la ruta de grabación, el lector 410 le pasa las fichas de los pasos grabados para grabación de datos nativa a la ficha pasante 435.

B. Ruta de Reproducción de la Grabadora Nativa

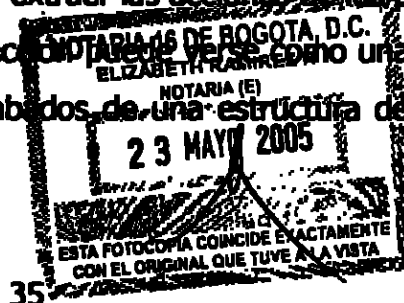
Para reproducción con la grabadora nativa de macros o UI, la herramienta de reproducción 420 pasa fichas (que se muestran como la ficha pasante 435) de la herramienta 420 a la aplicación 440 que incluye el ambiente de grabación nativo apropiado. La herramienta 420 le pasa la ficha a la interfase de reproducción



nativa especial. Antes de hacer esto, la herramienta 420 puede Interpretar capas de datos adicionales que fueron añadidas a las fichas de los pasos grabados, incluyendo información de marcación tal como el nombre de la aplicación, marcas de tiempo, valores de índice, etc., de tal manera que los pasos se reproducen en la aplicación correcta y en la secuencia apropiada.

A diferencia de otros tipos de Información en el código 450, las fichas de los pasos grabados en forma nativa no se interpretan directamente con la herramienta de reproducción 420. En cambio, las fichas se pasan como bloques de datos opacos por la ficha pasante 435 a la aplicación 440 (que se muestra como el componente que acepta las fichas 442). La aplicación 440 interpreta las fichas, pasando los datos de grabación de macros de la fichas al reproductor de macros 445. En la aplicación 440, los datos de macros grabados se Interpretan en el ambiente nativo de grabación de macros. Los datos de macros se expresan en un lenguaje tal como VBA, PerfectScript, M4, TeX u otro lenguaje macro. Las capacidades Internas de grabación de una aplicación suministran un marco de trabajo robusto y confiable para la reproducción, a veces permitiendo una reproducción que no es posible a través de la arquitectura de sistema UI.

La aplicación 440 extrae los pasos grabados preparándolos para la reproducción. La aplicación 440 puede extraer un paso grabado de una solo ficha, o la aplicación 440 puede extraer múltiples pasos de acciones grabadas de una solo ficha. En cuanto a la mecánica de la conversión a fichas en orden inverso, la aplicación 440 puede leer datos de acciones de pasos grabados de uno o más campos de la estructura de datos para la ficha, que pasó de la herramienta 420 como una copia por referencia. O, la aplicación 440 puede desempacar datos de acciones de pasos grabados que pasa como parámetros de una llamada de método. O, la aplicación 440 puede utilizar otro mecanismo para extraer las acciones de los pasos grabados como fichas. Conceptualmente, la extracción puede verse como una lectura de los datos de las acciones de los pasos grabados de una estructura de datos en una



ficha, o desempacando los datos de acciones de pasos grabados de la información de la ficha. Esto "revela" las instrucciones del lenguaje macro, que son reconocibles en el ambiente nativo de grabación apropiado.

La aplicación 440 puede proceder entonces a marcar los datos anexos o añadidos a los datos de las acciones de pasos grabados cuando se generan las fichas de los datos. La marcación de los datos suministra detalles pertinentes a la reproducción, por ejemplo, Información de la aplicación (por ejemplo el nombre de la aplicación y su versión), una marca de tiempo para indicar el orden, Información sobre privilegios (por ejemplo, que derechos se tienen para reproducir los pasos), etc.

La herramienta 420 es compatible con una interfase especial de reproducción nativa en el sentido de que formatea la información para pasarla a la aplicación 440 a través de la interfase, hace las llamadas apropiadas a través de la interfase, implementa métodos para regresar las llamadas para la aplicación 440, etc. La aplicación 440 es compatible con una interfase en el sentido de que implementa métodos para que la herramienta 420 llame, manipule información que le ha sido pasada por la herramienta, invoca los métodos para regresar las llamadas de la herramienta 420, etc. El mecanismo de procesamiento y de llamadas utilizado por la interfase dependen de la implementación.

C. Ruta de Reproducción del Sistema UI

La aplicación 440 con su grabadora reproductora de macros nativos y/o otras aplicación 460 pueden apoyar la reproducción de acuerdo a la arquitectura del sistema UI con el reproductor de la arquitectura del sistema UI 425. El reproductor 425 es en esencia un motor para controlar en forma programática rasgos de una aplicación basada en UI. Para esta ruta de reproducción, la herramienta de reproducción 420 procesa el código 450 con el reproductor 425 para reproducir los pasos grabados y la actividad UI a través de interacciones programáticas con la aplicación basada en UI.

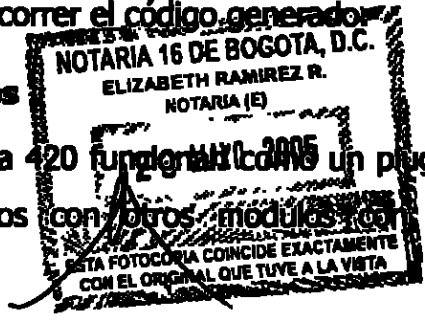
En algunas implementaciones, la herramienta 420 ejecuta una reproducción en la arquitectura del sistema UI con componentes tal como se describe en la solicitud de patente de E.U. llamada "Herramientas de Grabación/Reproducción para Aplicaciones basadas en UI", entregada el 15 de Junio de 2004. En forma alterna, la herramienta 420 utiliza otra técnica para reproducir con la arquitectura del sistema a UI.

Al inicio de la reproducción, una comparación se podría hacer entre las condiciones existentes y las condiciones bajo los cuales la grabación UI se hizo. Si se detectan diferencias significativas, se le informa al usuario, y se lo ofrece la opción de continuar.

En algunos casos, el código de reproducción se ejecuta por sí mismo y no requiere entradas o acciones de control adicionales. En forma alternativa, las acciones del sistema UI se ponen en una lista de espera para reproducción, que le permite al módulo de reproducción 425 procesar cada paso grabado en orden. El reproductor 425 obtiene un paso grabado de la lista, espera que la aplicación adecuada esté disponible, espera que el proceso esté listo para las entradas, ejecuta los pasos grabados, reporta el resultado de los pasos a la herramienta de reproducción 420 o en algún otro lado. Si cualquiera de esos pasos falla, una función simple de dormir se invoca, el reproductor 425 simplemente espera un período de tiempo determinado antes de intentar reproducir los pasos grabados de nuevo. O, cuando un paso no se puede ejecutar, se notifica al usuario y se le pide que complete paso. Esto permite que fallas en la reproducción se puedan manejar sin que Interrumpan a la herramienta de reproducción o cualquier otra aplicación asociada. En vez de depender en el reproductor 425, algunas aplicaciones lo pueden sustituir con su propia herramienta de reproducción para correr el código generado.

D. Arquitectura de Herramientas y Usos

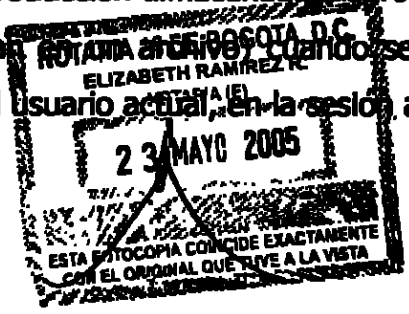
En general, los módulos varios de la herramienta 420 funcionan como un plug-In, lo que significa que pueden ser reemplazados con otros módulos con una



funcionalidad similar para distintas aplicaciones. La herramienta 420 por lo tanto es extensible; su funcionalidad se puede alterar para distintas aplicaciones. La herramienta 420 presenta un menú de opciones para módulos que funcionan como un plug-in, permitiendo de esta manera al usuario controlar la carga de módulos apropiados para la funcionalidad deseada.

La herramienta de reproducción suministra varias opciones que se pueden configurar. La herramienta de reproducción 420 puede configurarse para indicar como se debe ejecutar la reproducción. Por ejemplo, la herramienta de reproducción se puede configurar para qué pause durante la reproducción cuando un botón de "Pausa" se activa. O, la herramienta 420 pausa cuando se requiere la entrada de un usuario para continuar procesando el código de reproducción. La herramienta de reproducción se puede configurar para ejecutar reproducciones tan rápido como el sistema de computación corriendo el código pueda "Pintar". En forma alternativa, la reproducción ocurre tan rápido como es posible sin esperar a que se complete la Pintura. En algunas Implementaciones, series múltiples de reproducción se corren en forma simultánea contra aplicaciones separadas.

Para mejorar la seguridad de la reproducción, la herramienta de grabación 220 de la figura 2 puede codificar el código de reproducción y/o las fichas Incrustadas en él. Si es el caso, la herramienta de reproducción 420 codifica el código y/o las fichas. Otras medidas de seguridad incluyen el requerimiento de un nombre y una contraseña que el usuario debe Ingresar para la reproducción, no registrar estos datos, chequear la modalidad de seguridad del código de reproducción contra el modo de seguridad del usuario (por ejemplo, requiriendo el status del administrador al usuarios que trate de reproducir el gulón a nivel del administrador), y únicamente permitiendo reproducción almacenada (reproducción de un código y/o fichas que no se encuentran en el archivo). Cuando se puede verificar que el contenido del buffer lo grabó el usuario actual, en la sesión actual.



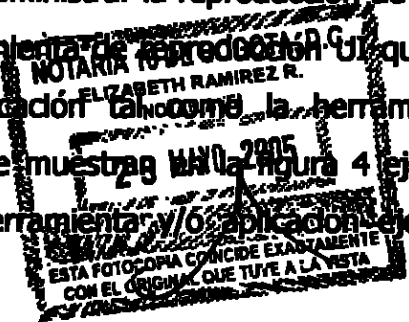
La herramienta 420 es responsable por cargar otros módulos plug-In en la medida que se necesiten, chequeando la seguridad (por ejemplo, chequeando privilegios), desovando secuencias de reproducción, sincronizando actividades entre módulos, construyendo y manteniendo colas utilizadas para reproducción, registrando errores (por ejemplo, a un archivo local u opcionalmente a un servidor de red o servicios), y recolectando información acerca de la reproducción.

La relación que se muestra entre módulos en la figura 4 Indica el flujo principal de Información; otras relaciones no se muestran por simplicidad. Dependiendo en la implementación, los módulos se pueden añadir, omitir, dividir en múltiples módulos, combinar con otros módulos, y/o reemplazar con módulos parecidos. En forma alternativa, una herramienta con distintos módulos y/o otras configuraciones de módulos ejecutan una o más de las técnicas de reproducción descritas acá.

La interfase de usuario de la herramienta 420 puede ser una Interfase del estilo VCR localizada alrededor de la ventana de reproducción, anclada en la parte superior o inferior de la pantalla del computador, o minimizada en la barra de tareas. La Interfase incluye botones para acciones como "Pausa", "Detener", y "Reproducir", como también un área para presentar descripciones de los pasos que se reproducen cuando esta descripción está disponible. Como ejemplo de tal interfase, ver la solicitud de patente de E.U. llamada "Herramientas de Grabación/Reproducción para Aplicaciones basadas en UI".

E. Técnicas de Reproducción

La figura 5. muestra una técnica 500 para administrar la reproducción de actividad macro nativa a la aplicación con una herramienta de reproducción UI que abarca todo el sistema. Una herramienta y aplicación tal como la herramienta de reproducción 420 y la aplicación 440 que se muestran en la figura 4 ejecutan la técnica 500. En forma alternativa, otra herramienta y/o aplicación ejecutan la técnica 500.



Handwritten signature or initials.

92

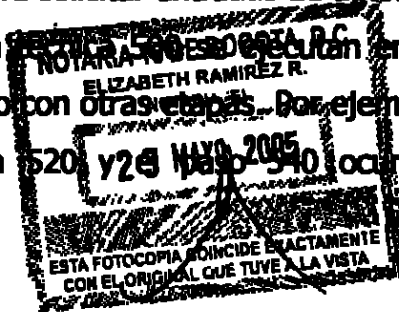
La herramienta recibe el código 510 con las fichas de los pasos grabados. Por ejemplo el código es un archivo de guión que incluye fichas. O, el código es un código en lenguaje de programación que tiene fichas incrustadas.

La herramienta luego verifica 520 el código. La herramienta puede verificar si las fichas son correctas y tienen el orden y la consistencia interna, o verifica que la herramienta y la aplicación estén listas para una reproducción correcta. Haciendo esto, la herramienta puede considerar marcar la información añadida a las fichas para verificar la identificación con las secuencias. Por ejemplo, la herramienta utiliza el mecanismo de verificación descrito anteriormente. Si una acción grabada no se puede verificar, se le pide al usuario que termine la reproducción o, alternativamente, ejecute una acción.

La herramienta luego pasa las fichas del paso grabado 540 a la aplicación apropiada. La aplicación procesa 550 las fichas, extrayendo datos de macros grabados, y reproduciendo las acciones de macro grabadas en el ambiente nativo de grabación y reproducción de la aplicación para reproducir los pasos ejecutados por un usuario contra la aplicación. Procesar las acciones grabadas en forma nativa incluye la reproducción de la actividad UI grabadas tan suave como sea posible con muy poca o ninguna interferencia por el usuario. Así, una herramienta de grabación del sistema UI puede efectivamente administrar e inter operar con el ambiente de grabación interno de la aplicación con un alto grado de confiabilidad.

La herramienta o aplicación determina 560 si continúa la reproducción. Por ejemplo, la herramienta y la aplicación terminan cuando el código con las fichas se han terminado de reproducir, cuando el usuario termina la reproducción, o cuando la reproducción termina debido a un error, o la herramienta y aplicación esperan cuando la reproducción se detiene para solicitar entradas de datos del usuario.

De otra manera, varias etapas de la secuencia 500 se ejecutan en forma separada con varias combinaciones en conjunto con otras etapas. Por ejemplo, mientras que la figura 5. muestra la verificación 520 y 540 ocurriendo antes del



193

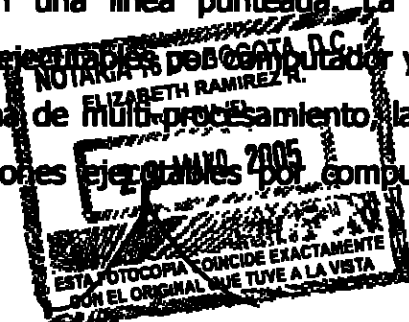
procesamiento de las fichas 550, alternativamente una o ambas de estas operaciones se ejecutan en forma concurrente con el proceso 550 para fichas en línea de reproducción.

IV. Ambiente de Computación

Las herramientas descritas anteriormente para grabación y reproducción de macros y UI para software diverso basado en GUI (por ejemplo, la herramienta 200 de la figura 2) y macros y técnicas de grabación UI se pueden implementar en una variedad de dispositivos de cómputo y ambientes, incluyendo computadores de varias formas (personal, estaciones de trabajo, servidores, de mano, laptops, tabletas u otros móviles), redes de computación distribuidas, servicios Web, como algunos ejemplos generales. La herramienta y técnicas de macro y grabación UI se pueden implementar en un circuito de hardware, como también sin grabaciones de macro, software de procesamiento y visualización 980 ejecutándose dentro de un computador u otros ambientes de computación, tales como los que se muestran en la figura 6.

La figura 6. muestra un ejemplo generalizado de un ambiente de computación adecuado 600 en donde las técnicas descritas se pueden implementar. El ambiente de computación 600 no pretende sugerir ninguna limitación en cuanto al alcance ni funcionalidad de las herramientas y técnicas acá descritas, pues ellas se pueden implementar en diversos ambientes de computación de propósito general o específico.

Con referencia a la figura 6, el ambiente de computación 600 incluye por lo menos una unidad de procesamiento 610 y una memoria 620. En la figura 6, esta configuración básica 630 se incluye con una línea punteada. La unidad de procesamiento 610 ejecutan instrucciones ejecutables por computador y puede ser un procesador real o virtual. En un sistema de multi-procesamiento, las unidades de multi-procesamiento ejecutan instrucciones ejecutables por computador para



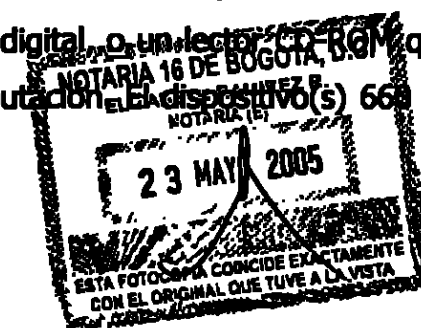
194

aumentar el poder de procesamiento. La memoria 620 puede ser una memoria volátil (por ejemplo, registros, cache, RAM), o memoria no volátil (por ejemplo, CD ROM, EEPROM, memoria flash, etc.) o alguna combinación de las dos. La memoria 620 almacena software 680 que implementa las herramientas y técnicas de grabación inteligentes.

Un ambiente de computación puede tener rasgos adicionales. Por ejemplo, el ambiente de computación 600 incluye el almacenaje 640, uno o más dispositivos de entrada 650, uno o más dispositivos de salida 660, y una o más conexiones de comunicación 670. Un mecanismo de interconexión (que no se muestran) tal como un bus, un controlador, o una red que interconecta los componentes del ambiente de computación 600. En forma típica, el software del sistema que opera (no se muestran) suministra un ambiente operativo para que otros software se ejecuten en el ambiente de computación 600, y coordina las actividades de los componentes en el ambiente de computación 600.

El almacenaje 640 puede ser removible o no removible, e incluye discos magnéticos, cintas magnéticas o casetes, CD-ROMs, CD-RWs, DVDs, o cualquier otro medio que se pueda utilizar para almacenar información y que se pueda acceder dentro del ambiente de computación 600. El almacenaje 640 almacena instrucciones para la grabación macro o UI y el software de reproducción 680.

El dispositivo(s) 650 (por ejemplo, para dispositivos que funcionan como un punto de control en la arquitectura de conectividad de los dispositivos) puede ser un dispositivo táctil de entrada como un teclado, un ratón, un bolígrafo, un trackball, un dispositivo de entrada de voz, un escáner, u otros dispositivos que puedan proveer entradas al ambiente de computación 600. Para audio, el dispositivo(s) de entrada 650 puede ser una tarjeta de sonido o dispositivo similar que acepte entradas de audio en forma analógica o digital, o un lector CD-ROM que suministra pruebas de audio al ambiente de computación 600. El dispositivo(s) 660 puede ser un



195

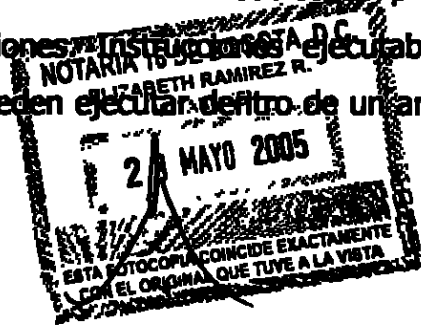
105

monitor, impresora, parlantes, escritor de CDs u otros dispositivo que provea salidas al ambiente de computación 600.

Las conexiones de comunicación 670 permiten la comunicación a través de un medio de comunicación con otra entidad de computación. El medio de comunicación suministra información tal como instrucciones ejecutables por computador, audio/vídeo u otros medios de información, u otros datos en señal de datos modulada. Una señal de datos modulada es una señal que tiene una o más de sus características fijas o alteradas de tal manera que codifica la información en la señal. Como ejemplo, y no como limitante, un medio de comunicación incluye técnicas alambradas e inalámbricas implementadas con transmisores tales como eléctricos, ópticos, infrarrojos, RF, acústicos y demás.

Las técnicas de macro y grabación y reproducción UI que se muestran acá pueden describirse en el contexto general de medios legibles por computador. Medios legibles por computador son cualquier medio disponible que se pueda acceder dentro de un ambiente de computación. Como ejemplo, y no como limitante, en el ambiente de computación 600, los medios legibles por computador incluyen la memoria 620, el almacenaje 640, los medios de comunicación, y una combinación de cualquiera de los antedichos.

Las técnicas presentadas se pueden describir en el contexto general de instrucciones ejecutables por computador, tales como las incluidas en módulos de programas, que se ejecutan en un ambiente de computación en un procesador real o virtual. Generalmente, los módulos de programa incluyen rutinas, programas, bibliotecas, objetos, clases, componentes, estructuras de datos, etc. que ejecutan distintas tareas o implementa tipos de datos abstractos. La funcionalidad de los módulos de programa se puede combinar o dividir entre módulos de programa como se desee en las varias implementaciones. Las instrucciones ejecutables por computador para módulos de programa se pueden ejecutar dentro de un ambiente de computación local o distribuido.

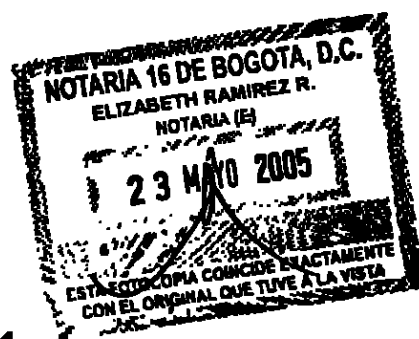


196

106

Para claridad en la presentación, la descripción detallada utiliza términos como "determinar", "generar", "ajustar", y "aplicar" para describir operaciones de computador en un ambiente de computación. Éstos términos son abstracciones de alto nivel para operaciones ejecutadas por un computador, y no se deben confundir con actos ejecutados por un ser humano. Las operaciones de computador actuales correspondientes a estos términos varían dependiendo de la implementación.

Como existen muchas implementaciones posibles en las cuales los principios de mi Invento se pueden aplicar, yo reivindicó dichas implementaciones como mi Invento en la medida que entren en el alcance y el espíritu de las siguientes reivindicaciones.



107

Reivindicamos:

1. Un medio legible por computador que almacena instrucciones legibles por computador para hacer que un sistema de computación programado consecuentemente ejecuta un método que consta de:

recibir una pluralidad de acciones Internas de macro que pasan en forma opaca como una o más fichas de pasos grabados de una aplicación basada en una interfase gráfica de usuario, y grabando uno o más pasos opacos grabados como fichas con un grabador externo UI.

2. El medio legible por computador de la reivindicación 1 donde una pluralidad de acciones de macro internas se generan por una grabadora Interna de macros de la aplicación y donde una o más fichas de pasos opacos grabados las puede manipular una grabadora externa UI que Interopera con la grabadora interna de macros de la aplicación.

3. El medio legible por computador de la reivindicación 1 comprende además:

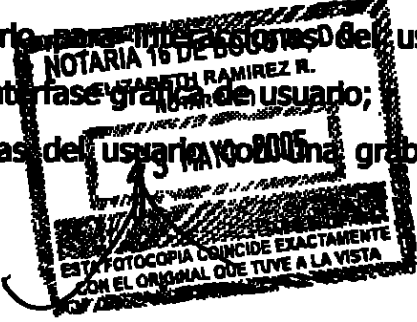
recibir una pluralidad de acciones de sistema en la Interfase del usuario para Interacciones del usuario contra una segunda aplicación basada en una Interfase gráfica de usuario;

generar una o más fichas de sistema de la Interfase de usuario para una pluralidad de acciones de sistema en la interfase del usuario; y

Grabar una o más fichas de sistema en la interfase del usuario con una grabadora externa a UI.

4. El medio legible por computador de la reivindicación 1 Incluye además: recibir una pluralidad de acciones directas del usuario para Interacciones del usuario contra una segunda aplicación basada en una Interfase gráfica de usuario;

grabando la pluralidad de acciones directas del usuario con una grabadora externa UI



198

5. El medio legible por computador de la reivindicación 1 comprende además la asociación de una o más fichas opacas de pasos grabados con datos relevantes en la grabadora externa UI.

6. El medio legible por computador de la reivindicación 5 donde los datos relevantes incluyen Información de Identificación de la aplicación para la aplicación.

7. El medio legible por computador de la reivindicación 1 donde una o más fichas opacas de pasos grabados son recibidas por una grabadora externa de una aplicación UI en modalidad de lotes.

8. El medio legible por computador de la reivindicación 1 donde una o más fichas opacas de pasos grabados son recibidas por una grabadora externa de una aplicación UI en modalidad interactiva.

9. El medio legible por computador de la reivindicación 1 donde la grabadora externa UI ínter opera con múltiples grabadoras internas de macro para distintas aplicaciones, donde las múltiples aplicaciones incluyen aplicaciones basadas en interfaces gráficas de usuario.

10. El medio legible por computador de la reivindicación 1 donde el método incluye además:

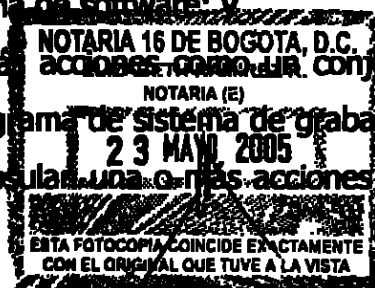
pasar una o más de las fichas opacas de pasos grabados de la grabadora externa UI devuelta a la aplicación; y

reproducir la pluralidad de acciones internas de macro.

11. Un medio legible por computador que almacena instrucciones ejecutables por computador para hacer que un sistema de computación programado concordantemente ejecute un método que comprende:

recibir un conjunto de una o más acciones expresadas en un primer lenguaje de grabación utilizado por un programa de software; y

representar el conjunto de una o más acciones como un conjunto de una o más fichas de grabación para un programa de sistema de grabación UI, donde una o mas fichas de grabación encapsulan una o más acciones de tal manera



199

que la una o más acciones son opacas dentro de la una o más fichas grabadas.

12. El medio legible por computador de la reivindicación 11 donde el programa de software genera una serie de una o más fichas de grabación.

13. El medio legible por computador de la reivindicación 11 donde el programa de grabación UI del sistema genera una serie de una o más fichas de grabación.

14. El medio legible por computador de la reivindicación 11 donde el método comprende además:

recibir un segundo conjunto de una o más acciones expresadas en un segundo lenguaje de grabación utilizado por un segundo programa de software; y
representar la segunda serie de una o más acciones como una segunda serie de una o más fichas de grabación para el programa de grabación UI del sistema.

15. El medio legible por computador de la reivindicación 11 donde el método comprende además:

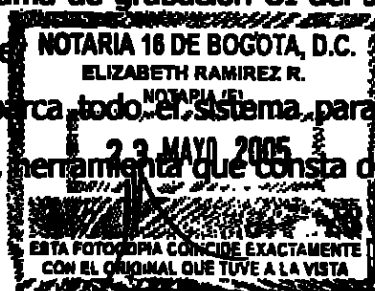
recibir un segundo conjunto de una o más acciones expresadas en el lenguaje de grabación UI del sistema; y

representar el segundo conjunto de una o más acciones como una segunda serie de una o más fichas de grabación para el programa de grabación UI del sistema.

16. El medio legible por computador de la reivindicación 11 donde una o más fichas de grabación son recibidas por el programa de grabación UI del sistema en modalidad de lotes del programa de software.

17. El medio legible por computador de la reivindicación 11 donde una o más fichas de grabación son recibidas por el programa de grabación UI del sistema en modalidad interactiva del programa de software.

18. Una herramienta de grabación UI que abarca todo el sistema para grabar y reproducir actividad en la interfase del usuario, herramienta que consta de:



un módulo para recolectar un primer conjunto de datos asociados con la actividad en la interfase del usuario contra una pluralidad de distintas aplicaciones, donde el primer conjunto de datos se expresa en una pluralidad de lenguajes de grabación nativos para una pluralidad de aplicaciones diferentes;

un módulo para manipular por lo menos parte del primer conjunto de datos recogidos como un conjunto de fichas; y

un módulo para administrar la reproducción de la actividad en la interfase del usuario suministrando fichas a la pluralidad de aplicaciones diferentes con los cuales el primer conjunto de datos se asocia.

19. Las herramientas de la reivindicación 18 comprenden además un módulo para convertir el conjunto de fichas en un código de salida.

20. Las herramientas de la reivindicación 18 donde los módulos son extensibles.

21. Las herramientas de la reivindicación 18 comprenden además:

un módulo para recoger un segundo conjunto de datos asociado con una segunda actividad de interfase de usuario, donde el segundo conjunto de datos se expresa en un lenguaje de sistema de grabación UI no nativo; y

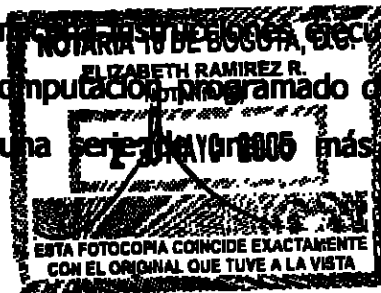
un módulo para administrar la reproducción de la segunda actividad en la interfase del usuario reproduciendo un segundo conjunto de datos.

22. La herramienta de la reivindicación 18 donde el primer conjunto de datos comprende instrucciones en lenguaje macro.

23. La herramientas de la reivindicación 18 donde la recolección del primer conjunto de datos es en modalidad de lotes.

24. La herramienta de la reivindicación 18 donde la recolección del primer conjunto de datos es en modalidad interactiva.

25. Un medio legible por computador que almacene instrucciones ejecutables por computador para hacer que un sistema de computación, programado de acuerdo ejecute un método que comprende: pasar una serie de fichas de 1005 más fichas de



~~201~~

201

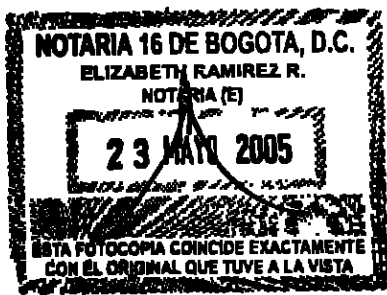
grabación de un programa de grabación UI de sistema a un programa de software para reproducción por una grabadora interna del programa de software; y extraer una serie de una o más acciones de grabación internas de una serie de una o más fichas de grabación, donde la serie de una o más acciones de grabación internas se expresan en un lenguaje de grabación interno utilizado por el programa de software, donde el programa de grabación UI del sistema inter opera con la grabadora interna para el programa de software.

26. El medio legible por computador de la reivindicación 25 donde el método comprende además:

pasar una segunda serie de una o más fichas de grabación del programa de grabación UI del sistema a un segundo programa de software para reproducir por una segunda grabadora interna de un segundo programa de software; y extraer una segunda serie de una o más acciones de grabación internas de una segunda serie de una o más fichas de grabación, donde la segunda serie de una o más acciones de grabación internas se expresan en un segundo lenguaje interno de grabación utilizado por un segundo programa de software, donde el programa de grabación UI del sistema interopera con una segunda grabadora interna para el segundo programa de software.

27. El medio legible por computador de la reivindicación 25 donde el método comprende además:

reproducir una segunda serie de una o más fichas de grabación para controlar programáticamente un segundo programa de software, donde una segunda serie de una o más fichas de grabación se expresan en lenguaje de grabación UI del sistema, donde el programa de grabación UI del sistema inter opera con un segundo programa de software.

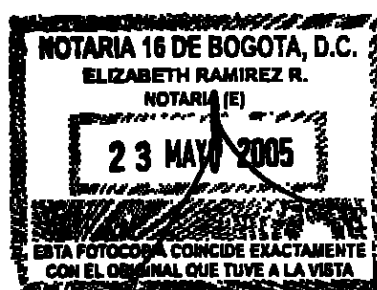


202

252

Resumen

Se describen técnicas y herramientas para grabar y reproducir en varias aplicaciones y en forma confiable actividad en la interfase gráfica del usuario ("GUI"), incluyendo aplicaciones que tienen una interfase de usuario dinámica. Las técnicas y herramientas se pueden utilizar en combinación o en forma separada. Por ejemplo, una herramienta de grabación recibe datos internos de macro de una aplicación basada en GUI como fichas opacas que están incrustadas en el código de reproducción. Una herramienta de reproducción ejecuta el código de reproducción pasando las fichas Incrustadas devuelta a la aplicación original para reproducirse en la grabadora Interna de la aplicación.



253

203

Traducción Oficial

REGISTRO DE ABOGADO No. 3382-87840-01	DECLARACIÓN Y PODER DE ABOGADO PARA SOLICITUD DE PATENTE	Expediente MS No. 308158.01
---	---	---------------------------------------

En mi calidad de inventor, por medio del presente documento declaro que:

Mi dirección de residencia/ dirección postal y nacionalidad aparecen consignados más abajo en este documento, enseguida de mi nombre;

Afirmo ser el inventor original, primero y único (si hay un solo nombre en la lista), o uno de los inventores originales y colectivo (si se encuentran varios nombres en la lista) del asunto que se reivindica y para el cual se solicita la patente del invento titulado: **SMART UI RECORDING AND PLAYBACK FRAMEWORK. MARCO DE TRABAJO PARA LA GRABACIÓN Y REPRODUCCIÓN UI INTELIGENTE**

Cuyas especificaciones se presentan con este documento, a menos que esté indicado o marcado en el paréntesis que se encuentra a continuación:

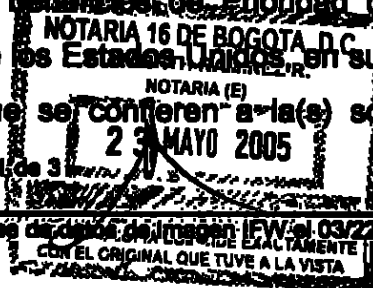
- ☐ Se radicaron el _____ con el número de serie para Solicitud de Patente en Estados Unidos o Solicitud Internacional PCT Número _____ y fueron modificadas el _____ (si aplica).

Declaro que he revisado y tengo pleno conocimiento y entendimiento de las especificaciones arriba mencionadas, incluyendo las reivindicaciones y modificaciones a las que se hace referencia arriba. Reconozco el deber de revelar y publicar toda la información que se considera material útil para otorgar la patente según lo que se define en el 37 del Código CFR 1.56.

Solicitudes Internacionales y/o Petición de Prioridad de Extranjería

Con el presente documento solicito los beneficios de Prioridad de Extranjería establecidos en el Título 35 del Código de los Estados Unidos, en su Artículo 119 (*United States Code, Section 119*) y que se confieren a la(s) solicitud(es) de

Página 1 de 3



Copia Suministrada por el USPTO de la Base de datos de Imágenes IFW el 03/22/2005

204

204

Patente(s) a nivel Internacional o a los certificados de inventor(es) que se relacionan e identifican en la lista a continuación, como también a las solicitudes de inscripción de Patentes Internacionales o de Certificados de Inventor(es) que se hayan radicado con anterioridad a la fecha en que se solicita la prioridad :

PAIS	SOLICITUD NÚMERO	FECHA DE RADICACIÓN	PRIORIDAD SOLICITADA BAJO EL TÍTULO 35 DEL ARTÍCULO 119 DEL U.S.C.
			SI: ☐ NO: ☐
			SI: ☐ NO: ☐

PODER DE REPRESENTACIÓN DE ABOGADO:

En mi calidad de inventor, en el presente documento designo al (los) siguiente(s) abogado(s) y/o representante(s) asociados con

El Número de Cliente 26119

Para que realice todas las gestiones relacionadas con esta solicitud ante la Oficina de Patentes y Marcas Registradas.

Enviar Correspondencia a:	Llamar directamente a:
El Número de Cliente 26119 Stephen A. Wight Klarquist Sparkman, LLP 121 S.W. Salmon Street, Suite 1600 Portland, Oregon 97204	Stephen A. Wight (503) 226-7391

En el presente documento declaro que todas estas afirmaciones las hago bajo mi pleno conocimiento y son verdaderas, además que toda la información que suministro es cierta; como también que todas estas aseveraciones las hago sabiendo que las afirmaciones falsas que se hacen de manera intencional son consideradas de carácter penal y se castigan con multas o prisión o de ambas formas, según lo establecido bajo el Título 18 del Código de los Estados Unidos en su Artículo 1001 y que dicho tipo de afirmaciones falsas ponen en peligro la validez de la solicitud o cualquier patente otorgada a partir de ellas.

Página 2 de 3

Copia Suministrada por el USPTO de la Base de Datos de Información. F.W. 1003/22/2005

205

205

REGISTRO DE ABOGADO No. 3382-67840-01	DECLARACIÓN Y PODER DE ABOGADO PARA SOLICITUD DE PATENTE	Expediente MS No. 308158.01
--	---	--

Nombre Completo del Inventor:

Howard B. Herdeg, III

Ciudadano de :

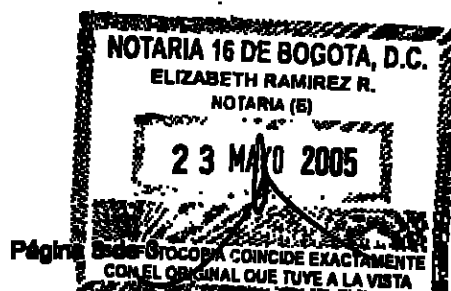
Estados Unidos de América

Dirección Residencia: North Bend, WA

Dirección Postal: 44415 166th Street SE, North Bend, WA, 98045

Ilegible
Firma del Inventor

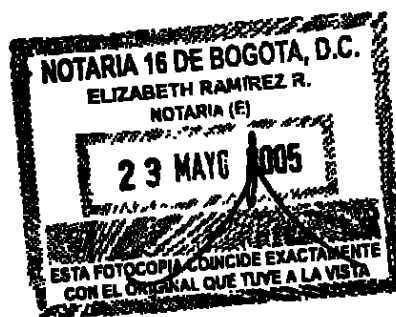
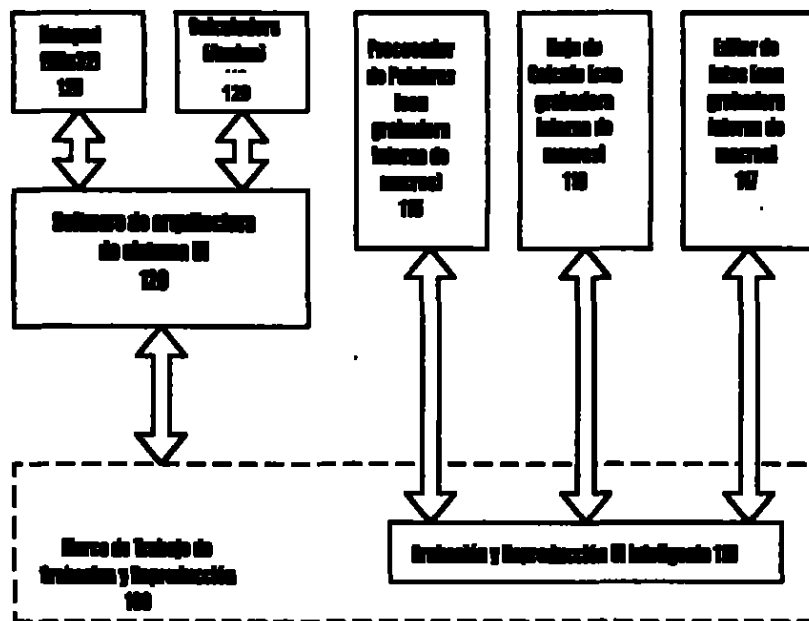
6/30/2004
Fecha



Copia Suministrada por el USPTO de la Base de datos de Imagen IFW el 03/22/2005

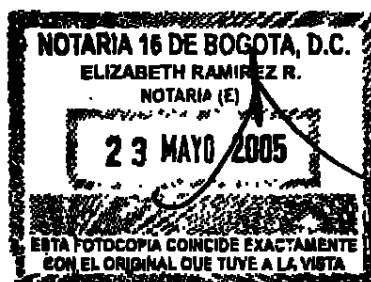
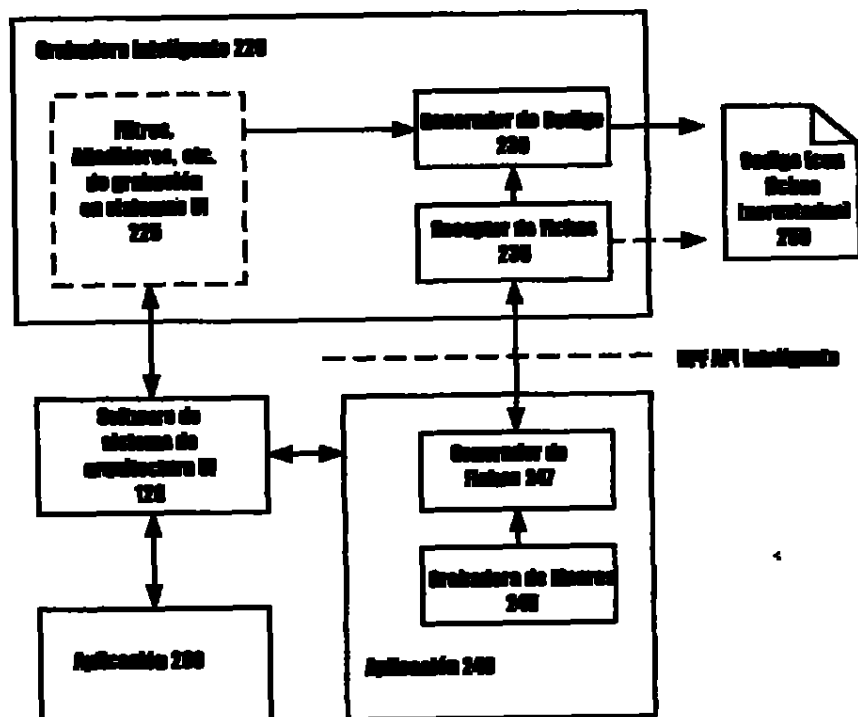
206

Figura 1



297

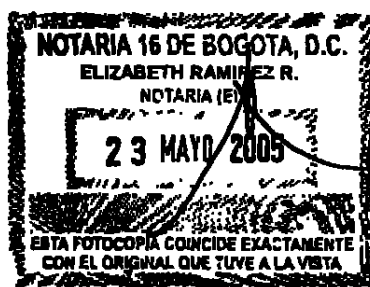
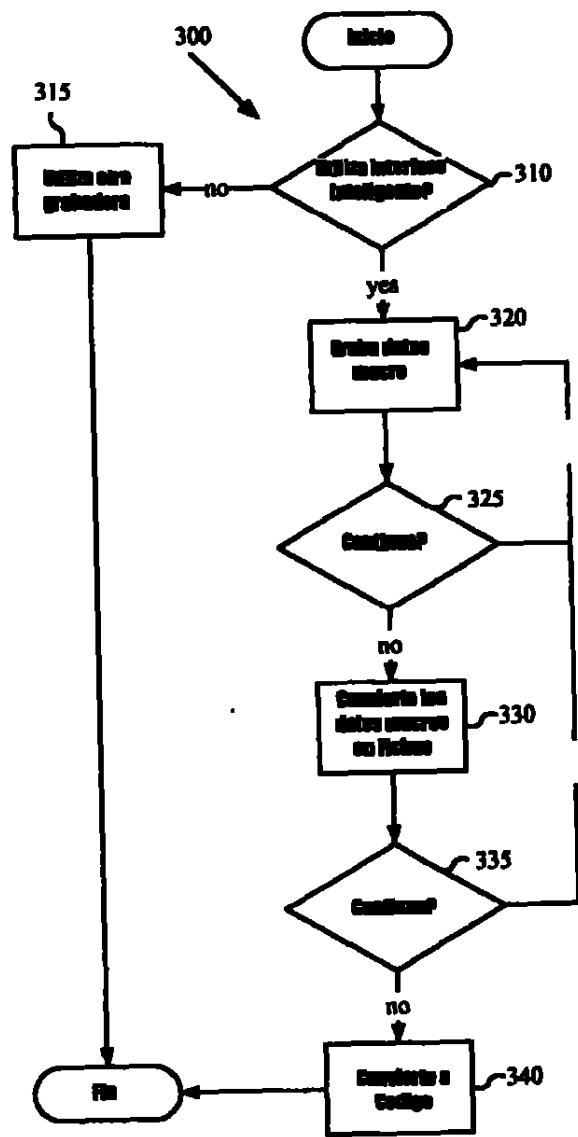
Figura 2



298

208

Figura 3



209

Figura 4

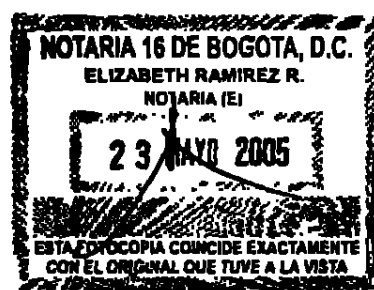
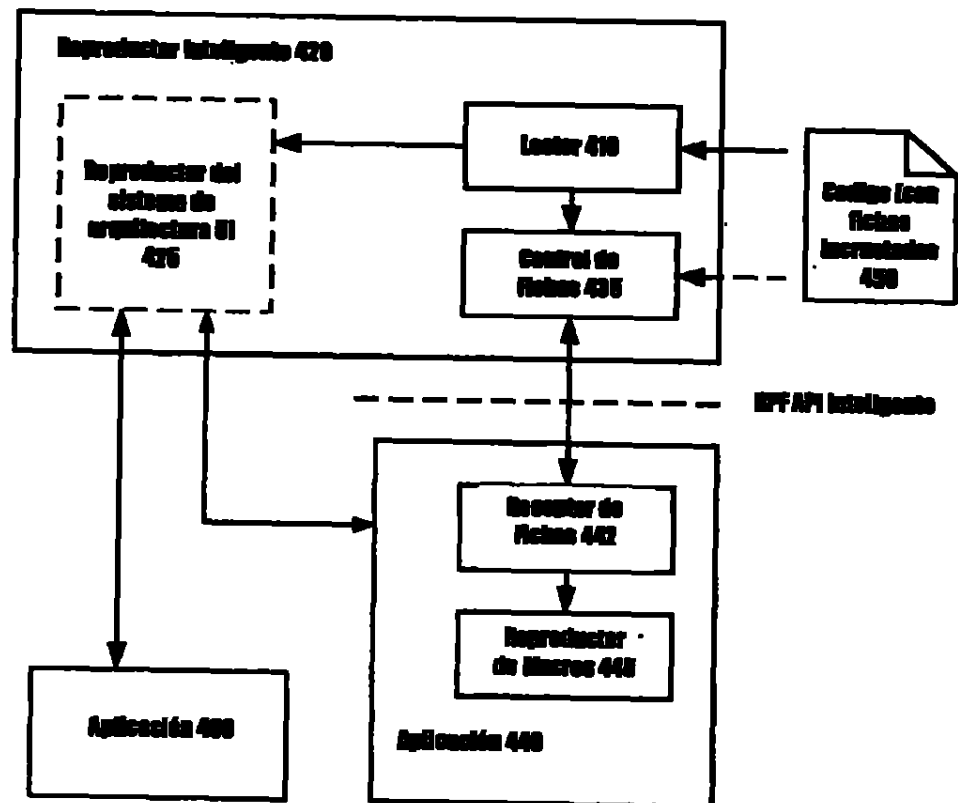


Figura 5

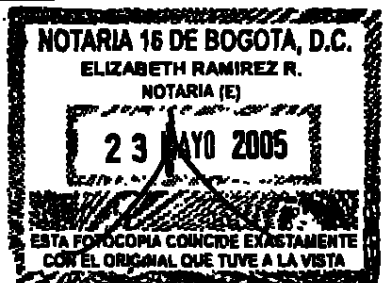
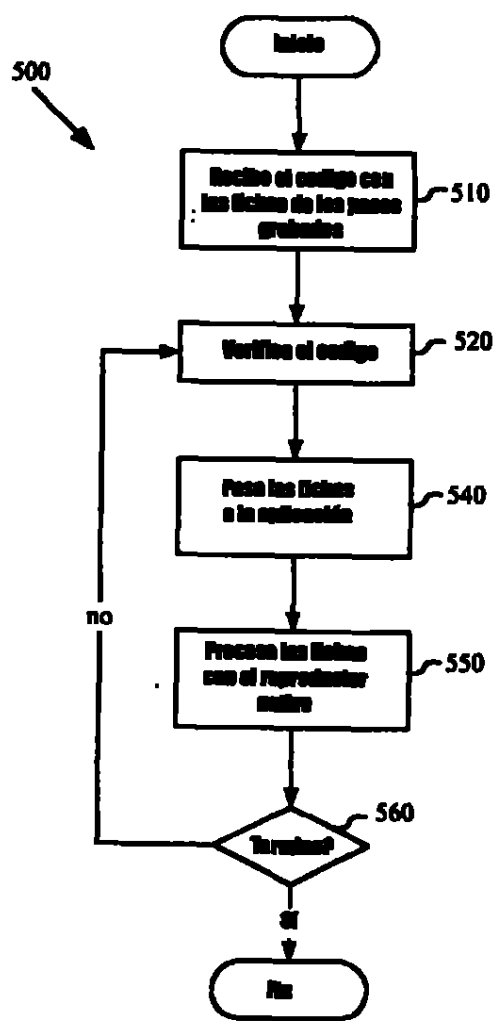
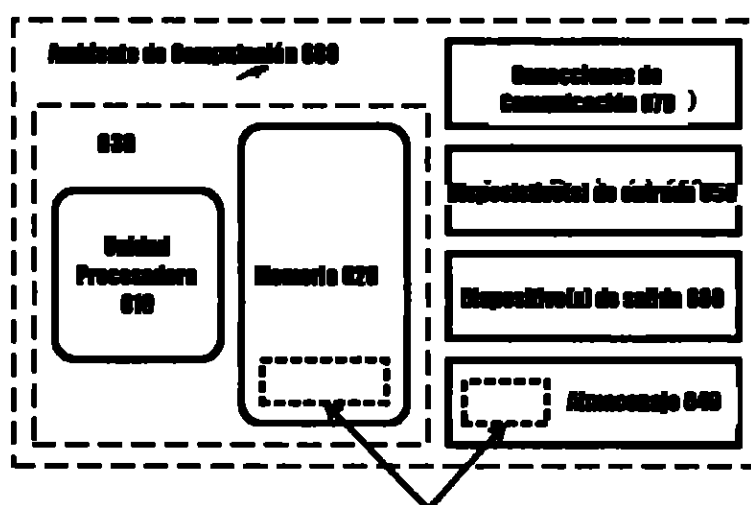
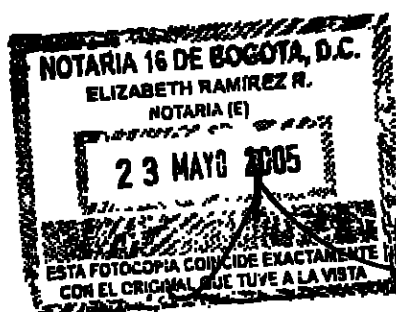


Figura 6



Software para grabación y reproducción inteligente 680



APOSTILLE

(Convention de La Haye du 5 octobre 1961)

212

1. Country: United States of America

This public document

2. has been signed by T. Lawrence

3. acting in the capacity of Certifying Officer

4. bears the seal/stamp of Patent and Trademark Office

Certified

5. at Washington, D.C.

6. the twenty-ninth of March, 2005

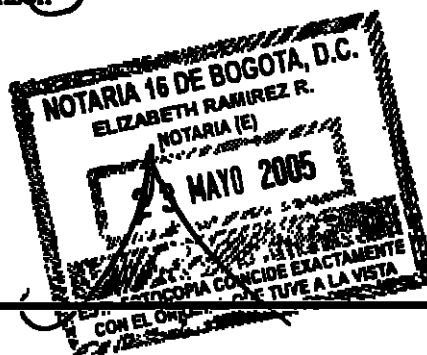
7. by Assistant Authentication Officer, United States Department of State

8. No. 05014915-12

9. Seal/Stamp:

10. Signature:

Sonya N. Johnson



A 1299120

THE UNITED STATES OF AMERICA

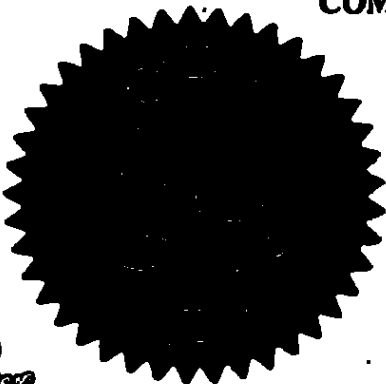
TO ALL TO WHOM THESE PRESENTS SHALL COME:

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office

March 25, 2005

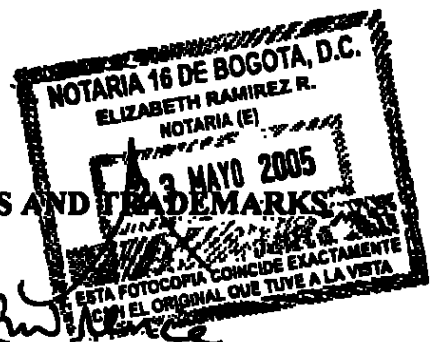
THIS IS TO CERTIFY THAT ANNEXED IS A TRUE COPY FROM THE
RECORDS OF THIS OFFICE OF A DOCUMENT RECORDED ON
JUNE 30, 2004.

By Authority of the
COMMISSIONER OF PATENTS AND TRADEMARKS



T. Lawrence

T. LAWRENCE
Certifying Officer



213

213

214

Form 06/2004 Document is ProLaw

07-13-2004

Atty. Ref. No. 3382-67840-01
press Mail Label No. EV 331582587 US

RECORDA1

102788744

PATENTS ONLY

1057 U.S. PTO
10/882861

COMMISSIONER FOR PATENTS

P.O. BOX 1450

ALEXANDRIA, VA 22313-1450 6-30-04

1. Total number of pages including cover sheet, attachments and documents: 3

2. Name of Conveying Party(ies) and Execution Date(s) of Document(s) ☐ Check here if additional name(s) attached

Name(s): Howard B. Herdag, III

Execution Date(s): June 30, 2004

3. Name and address of receiving party ☐ Check here if additional name(s) & address(es) are attached

Name Microsoft Corporation

Address One Microsoft Way

City Redmond

State/Country WA

1082861

Zip 98052

4. Nature of Conveyance

☒ Assignment

☐ Merger

☐ Security Agreement

☐ Name Change

☐ Other:

5. Total number of applications and patents involved: 1

6. Total Fee Enclosed (37 C.F.R. § 3.41): \$40.00

7. ☒ Check here if any deficiency/overpayment is authorized to be charged to deposit account 02-4550

8. Enter either the Execution date (of the Declaration and Power of Attorney), Application Number, or the Patent Number. Do not enter more than one number for the same patent.

A. ☒ This document is being filed with a new application. Execution date is: June 30, 2004

B. ☐ Patent Application No.(s) or Patent No.(s):

☐ Check here if additional numbers are entered

9. Correspondent's name, address, and telephone number

Kyle B. Rinehart

Klarquist Sparkman, LLP

One World Trade Center, Suite 1600

121 S.W. Salmon Street

Portland, Oregon 97204-2988

Telephone: 503-226-7391

Facsimile: 503-228-9446

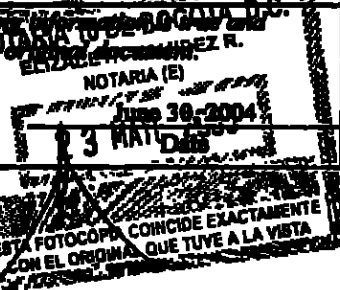
10. ☒ Please return the attached postcard to confirm that these items have been received.

11. Statement and signature To the best of my knowledge and belief, the foregoing is a true and correct copy of the original document.

Kyle B. Rinehart

Name of Person Signing

Signature



07/06/2004 NEWARK 00000079 10682861

14 FC:0021

40.00 CP

cc: Accounting (only if via fax)
Docketing

PATENT

REEL: 016647 FRAME: 0717

214

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Filing Date Executed concurrently herewith
Inventorship Herdeg
Applicant Microsoft Corporation
Attorney's Docket No. 3382-67840-01
MS Docket No. 308158.01
Title: **SMART UI RECORDING AND PLAYBACK FRAMEWORK**

PATENT ASSIGNMENT

PARTIES TO THE ASSIGNMENT

Assignor(s):

Howard B. Herdeg, III
44415 166th Street SE
North Bend, WA 98045
United States of America

Assignee:

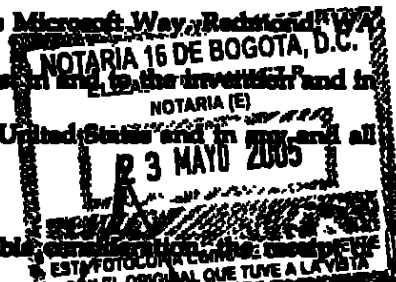
Microsoft Corporation
Corporation in the State of Washington
One Microsoft Way
Redmond, WA 98052

AGREEMENT

WHEREAS, ASSIGNOR(S) (listed above) are inventor(s) of an invention entitled "SMART UI RECORDING AND PLAYBACK FRAMEWORK," as described and claimed in the specification forming part of an application for United States letters patent referenced above;

WHEREAS, Microsoft Corporation (hereinafter referred to as ASSIGNEE), a corporation of the State of Washington having a place of business at One Microsoft Way, Redmond, WA 98052, is desirous of acquiring the entire right, title and interest in and to the invention and in and to any letters patent that may be granted therefor in the United States and in any and all foreign countries;

NOW, THEREFORE, in exchange for good and valuable consideration, the receipt of which is hereby acknowledged, ASSIGNOR(S) hereby sell, assign and transfer unto ASSIGNEE,



215

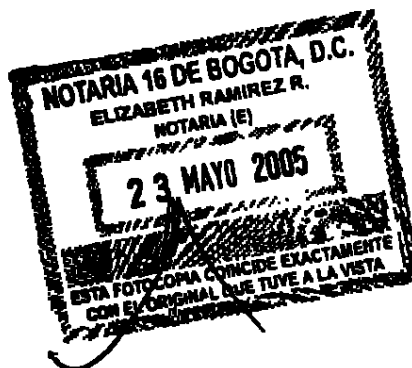
216

ASOAm 06/30/04 1303-07540-01 030812L01

the entire right, title and interest in and to said invention, said application and any and all letters patent which may be granted for said invention in the United States of America and its territorial possessions and in any and all foreign countries, and in any and all divisions, reissues and continuations thereof, including the right to file foreign applications directly in the name of ASSIGNEE and to claim priority rights deriving from said United States application to which said foreign applications are entitled by virtue of international convention, treaty or otherwise, said invention, application and all letters patent on said invention to be held and enjoyed by ASSIGNEE and its successors and assigns for their use and benefit and of their successors and assigns as fully and entirely as the same would have been held and enjoyed by ASSIGNOR(S) had this assignment, transfer and sale not been made. ASSIGNOR(S) hereby authorize and request the Commissioner of Patents and Trademarks to issue all letters patent on said invention to ASSIGNEE. ASSIGNOR(S) agree to execute all instruments and documents required for the making and prosecution of applications for United States and foreign letters patent on said invention, for litigation regarding said letters patent, or for the purpose of protecting title to said invention or letters patent therefor.

6/30/2004
Date

Howard B. Harding
Howard B. Harding



216

217

TRADUCCIÓN OFICIAL

APOSTILLA

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

1. País: Estados Unidos de América
2. El presente documento público ha sido firmado por: T. Lawrence
3. actuando en calidad de: Funcionario que Certifica
4. Lleva el sello / estampilla de: La Oficina de Patentes y Marcas Registradas / *Patent and Trademark Office*

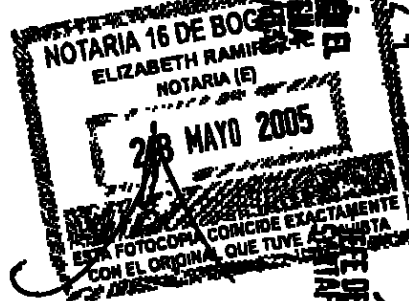
CERTIFICADO

5. En: Washington, D.C.
6. El: Veintinueve de Marzo de 2005
7. Por: El Funcionario Asistente de Autenticaciones, Departamento Estado de Los Estados Unidos.
8. Bajo el número: 05014915-12
9. Lleva Sello / Estampilla: Sello del DEPARTAMENTO DE ESTADO DE LOS ESTADOS UNIDOS DE AMÉRICA / *United States Department of State*
10. Firmado por: Sonya N. Johnson

Martha S. Samiento S.

Martha S. Samiento S.

Traductora Oficial Juramentada
Resolución No. 835 Mijusticia



MINISTERIO DE RELACIONES
EXTERIORES
Ref. Sonya
1729
2005 MAY 19 P 1:26
NO SE ASUME
RESPONSABILIDAD DEL TEXTO
OFICINA DE LEGALIZACIONES
ESTADO DE WISCONSINA D.C.

217

218

Traducción Oficial

LOS ESTADOS UNIDOS DE AMÉRICA

A TODOS AQUELLOS QUE LES PUEDA INTERESAR

**EL DEPARTAMENTO DE COMERCIO DE LOS ESTADOS UNIDOS
LA OFICINA DE PATENTES Y MARCAS REGISTRADAS DE LOS ESTADOS UNIDOS**

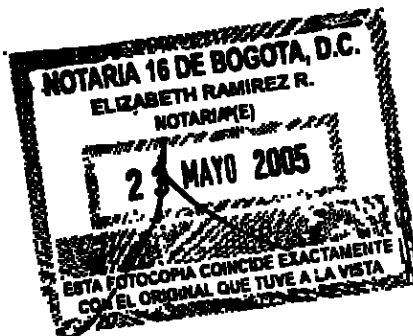
25 de Marzo de 2005

**CON ESTE CERTIFICAMOS QUE EL DOCUMENTO ANEXO ES COPIA
VERDADERA DE LOS REGISTROS Y ANOTACIONES REALIZADOS EN ESTA
OFICINA, DEL DOCUMENTO REGISTRADO EL 30 DE JUNIO DE 2004.**

**Autorizado por el
COMISIONADO DE PATENTES Y MARCAS REGISTRADAS**

**Firma: T. LAWRENCE
Funcionario Autorizado para Certificar**

**Sello Dorado: DEPARTAMENTO DE COMERCIO
OFICINA DE PATENTES Y MARCAS REGISTRADAS DE LOS ESTADOS UNIDOS**



Martha S. Laventure S.

**Martha S. Laventure S.
Traductora Oficial Juramentada
Resolución No. 835 Minjusticia**

218

219

Traducción Oficial

REGISTRO DE TRASPASO

Atty. Ref. No. 3382-87840-01

Etiqueta de Correo Expreso No. EV 331582587 US

JMO:icm 06-30-04 Documento en
ProLay

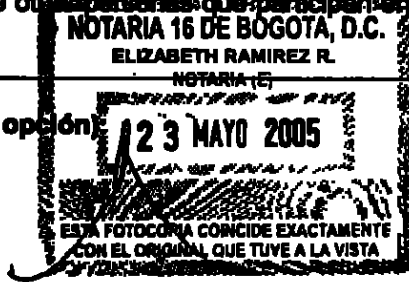
07 - 13 - 2004
Código de Barras
102788744

19587 U.S. PTO
10/882881
Código de Barras
083004

COMMISSIONER FOR PATENTS / COMISIONADO PARA PATENTES

P. BOX 1450

ALEXANDRIA, VA 22313-1450 6-30-04

1. Número total de páginas incluyendo la portada, los anexos y el documento: 3	
2. Nombre(s) de la(s) Parte(s) que Cede(n) la patente y fecha(s) de ejecución del(los) Documento(s): Nombre(s): Howard B. Herdeg, III Fecha(s) de Ejecución: 30 de Junio de 2004 ☐ Marque si se incluyen los nombres de otras personas que participan en este traspaso de patente.	
3. Nombre y dirección de la(s) parte(s) que reciben el traspaso de la patente: Nombre: Microsoft Corporation Dirección: One Microsoft Way Ciudad: Redmond Estado/Pais: Washington 10882861 Código Postal: 98052 ☐ Marque si se incluyen los nombres y las direcciones de otras personas que participan en este traspaso de patente.	
4. Carácter o naturaleza del traspaso (marque solo una opción): ☒ Traspaso ☐ Fusión ☐ Acuerdo/Póliza de Seguro ☐ Cambio de nombre ☐ Otro:	

219

Martha L. Sarmiento L.

Martha L. Sarmiento L.

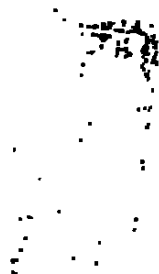
Traductora Oficial Juramentada
Resolución No. 895 Mijusticia



Martha S. Lamentado S.

Martha S. Lamentado S.

Traductora Oficial Juramentada
Resolución No. 835 Minjusticia



221

Traducción Oficial

**EN OFICINA DE PATENTES Y MARCAS REGISTRADAS
DE LOS ESTADOS UNIDOS**

Fecha de Radicación

**Se realiza de manera simultanea con el
documento adjunto**

Inventor

Herdeg

Solicitante

Microsoft Corporation

Expediente de Abogado No.

3382-67840-01

Número de Expediente MS.

308158.01

**Nombre: SMART UI RECORDING AND PLAYBACK FRAMEWORK / MARC
DE TRABAJO PARA LA GRABACIÓN Y REPRODUCCIÓN UI
INTELIGENTE**

TRASPASO DE PATENTE

PARTES QUE SUSCRIBEN ESTE TRASPASO

Cedente(s):

**Howard B. Herdeg, III
44415 166th Street SE
North Bend, WA, 98045
United States of America**

Beneficiario de la Patente:

**Microsoft Corporation
Corporation in the State of Washington /Corporación con sede en el Estado de
Washington
One Microsoft Way
Redmond, WA 98052**



221

Martha S. Llanusa S.

Martha S. Llanusa S.

Traductora Oficial Juramentada
Resolución No. 835 Minjusticia

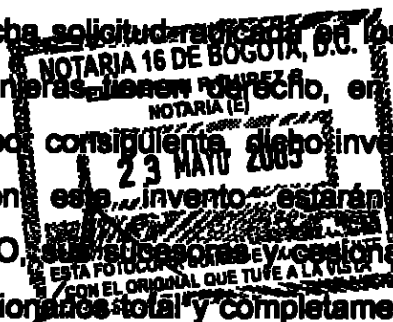
222

ACUERDO

CONSIDERANDO QUE EL(LOS) CEDENTE(S) (enunciado(s) anteriormente) es (son) el (los) creador(es) del invento denominado "SMART UI RECORDEN AND PLAYBACK FRAMEWORK" MARCO DE TRABAJO PARA LA GRABACIÓN Y REPRODUCCIÓN UI INTELIGENTE, patentado según la descripción y reivindicaciones que hacen parte de la solicitud de Registro de Patente para los Estados Unidos, arriba mencionada;

CONSIDERANDO QUE la empresa Microsoft Corporation (denominada de aquí en adelante el CESIONARIO), es una corporación del Estado de Washington, con sede en One Microsoft Way, Redmond, WA 98052, está interesada en adquirir en su totalidad los derechos, el título y los intereses relacionados con este invento y con el registro de patente que se le conceda en los Estados Unidos y en cualquier otro país;

POR CONSIGUIENTE, como resultado de esta negociación, por la presente se da reconocimiento al recibo fruto de esta transacción. Con la presente, el(los) CEDENTE(S) vende(n), asigna(n) y transfiere(n) al CESIONARIO la totalidad de los derechos, título e intereses relacionados con este invento, con la solicitud y con las patentes que se le otorguen a dicho invento en los Estados Unidos de América y en sus posesiones territoriales, o en otro país, y todas sus divisiones, reediciones y continuaciones de ésta, incluyendo el derecho a radicar solicitudes de patentes extranjeras directamente en nombre del CESIONARIO y solicitar los derechos de prioridad que se derivan de dicha solicitud radicada en los Estados Unidos por lo cual dichas solicitudes extranjeras tienen derecho, en virtud de Convenciones Internacionales o tratados y por consiguiente, dicho invento, o las solicitudes de patentes relacionadas con este invento, estarán bajo la responsabilidad y beneficio del CESIONARIO, sus sucesores y cesionarios para su uso y beneficio y de sus sucesores y cesionarios total y completamente, de la misma forma como el (los) CEDENTE(S) que tenía(n) esta asignación y



222

Martha S. Samiento S.

Martha S. Samiento S.

Traductora Oficial Juramentada
Resolución No. 835 Mijusticia

Martha S. Samiento S.

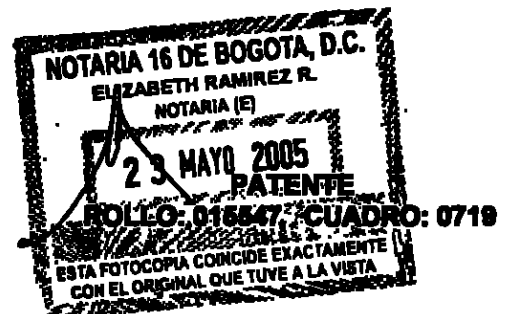
223

responsabilidad, la hubiera(n) mantenido y disfrutado si la venta y traspaso no se hubieran realizado. El(los) CEDENTE(S) por la presente autoriza(n) y solicita(n) al Comisionado de Patentes y Marcas Registradas que expida y otorgue todas las Patentes del invento al CESIONARIO. El(los) CEDENTE(S) se compromete(n) a ejecutar todos los instrumentos y documentos requeridos para diligenciar y procesar las solicitudes de patente relacionadas con dicho invento para los Estados Unidos y otros países, así como para litigación relacionada con dichos registros de patentes, o con el propósito de proteger el nombre, los derechos de dicho invento y su patente.

6/30/2004
Fecha

Firma ilegible
Howard B. Herdeg

REGISTRADO EL: 06/30/2004



223

Martha S. Samiento S.
Martha S. Samiento S.

Traductora Oficial Juramentada
Resolución No. 835 Minjusticia

744

224

CONSTANCIA DE TRADUCCIÓN OFICIAL

Yo, Martha Sofía Sarmiento Solano, identificada con Cédula de Ciudadanía No. 41.686.291, de Bogotá, HAGO CONSTAR que el presente documento es versión fiel y completa en idioma Español de los siguientes documentos:

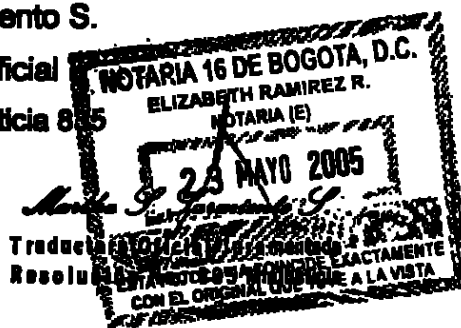
- Apostilla No. 05014915- 12 del 29 de Marzo de 2005 y firmada por Sonya N. Jonson, Funcionario Asistente de Autenticaciones del Departamento de Estado de Los Estados Unidos.
- Certificado del Departamento de Comercio de los Estados Unidos firmado por: T. Lawrence, Funcionario Autorizado para Certificar, con fecha 25 de Marzo de 2005.
- Registro de Traspaso de Patente, con fecha del 30 de Junio de 2004 y firmado por Kyle B. Rinehart.
- Traspaso de Patente y Acuerdo bajo el Expediente de Abogado No. 3382-67840-01 que se encuentra firmado por Howard B. Herdeg, inventor.

Martha S. Sarmiento S.

Martha S. Sarmiento S.

Traductora Oficial

Resolución Minjusticia 835



224

225
125

REQUISITOS NECESARIOS PARA PRESENTACION Y AGILIZACION DEL TRAMITE

	USUARIO	RADICADOR
PATENTE DE INVENCION	()	()
MODELO DE UTILIDAD	()	()
- Indicación de que se solicita la concesión de una patente.	()	()
- Datos de identificación del solicitante o de la persona que se presenta la solicitud.	()	()
- Descripción de la invención.	()	()
- Dibujos de ser estos pertinentes.	()	()
- Comprobante de Pago de las tasas establecidas.	()	()

COMPLETA () INCOMPLETA ()

DISEÑO INDUSTRIAL ()

- Indicación de que se solicita el registro de Diseño Industrial	()	()
- Datos de identificación del solicitante o de la persona que presenta la solicitud.	()	()
- Representación gráfica o fotográfica del Diseño Industrial o muestra del Material que incorpora el diseño.	()	()
- Comprobante de pago de las tasas establecidas	()	()

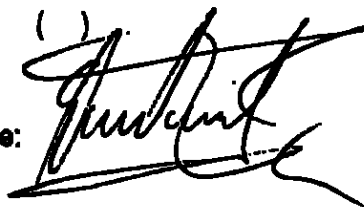
COMPLETA () INCOMPLETA ()

ESQUEMA DE TRAZADO ()

- Indicación de que solicita el registro de un Esquema de trazado.	()	()
- Datos de identificación del solicitante o de la persona que presenta la solicitud.	()	()
- Representación gráfica del esquema de trazado	()	()
- Comprobante de pago de las tasas establecidas.	()	()

COMPLETA () INCOMPLETA ()

Firma Responsable:



Fecha:

REQUISITOS NECESARIOS PARA PRESENTACION Y AGILIZACION DEL TRAMITE

226

~~226~~

PROTOCOLO

Expediente 05 - 51829

**SUPERINDUSTRIA Y COMERCIO
SISTEMA DE REGISTRO**

MANTENIMIENTO DE PROTOCOLOS

Número : 18089
Fecha : 04/08/1995
Conferido: MICROSOFT CORPORATION
País : US ESTADOS UNIDOS DE AMERICA
Ciudad : 234 REDMOND, WASHINGTON
Represent: S Sustituir: S Desistir: S
No. Radicación: 95 34920 Clase: 0 Seev: 0 Seac: 0

Indicia	Código	Ctr	Nombre
1	77	A	CARDENAS NAVAS DARIO
2	3653		CARDENAS CABALLERO EDUARDO
3	5048		CARDENAS MARTINEZ BERNARDO

Encontrados (1/1) registro(s)

Contestar favor indique el número de radicación consignado en el sticker

Sede Centro: Carrera 13 No. 27-00 Pisos 2, 3, 7 y 10
Commutador: 3 820840 Fax: 3 305220
Sede CAN: Tr. 40A No. 38-50 3133263 al 69
Fax: 3 305220. Línea 9800-910 163
Web: www.ic.gov.co e-mail: informa@ic.gov.co
Bogotá D.C. - Colombia

Folio 227

2020
Bogotá DC

Doctor(a)
DARIO CARDENAS NAVAS
Apoderado(a)

REFERENCIA:	Radicación No.	05 - 51829
	Trámite	002
	Evento	001
	Actuación	416
	Oficio No.	434
	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.

De acuerdo con lo previsto por el artículo 40 de la Decisión 486/00, la publicación en este caso se hará a partir del día 27 de noviembre de 2006.

Cúmplase
Dado en Bogotá DC, a los

30 JUL 2006


ALIX CARMENZA CÉSPEDES DE VERGEL
Jefe de la División de Nuevas Creaciones

Al contestar favor indique el número de radicación consignado en el sticker

Sede Centro: Carrera 13 No. 27-00 Pisos 2, 5, 7 y 10
Commutador: 3 820840 Fax: 3 505220
Sede CAN: Tr. 40A No. 38-50 3153265 al 69
Fax: 3 505220. Línea 9800-910 165
Web: www.ic.gov.co e-mail: info@ic.gov.co
Bogotá D.C. - Colombia