República Federativa do Brasil
Ministério da Indústria, Comércio Exterior
e Serviços
Instituto Nacional da Propriedade Industrial

(11) PI 0404008-2 B1

(22) Data do Depósito: 23/09/2004

(45) Data de Concessão: 28/03/2017

(54) Título: SISTEMA QUE GERENCIA O PARTICIONAMENTO DE UMA APLICAÇÃO, MÉTODO DE UM PRIMEIRO OBJETO DE SOFTWARE QUE EXECUTA EM UM PRIMEIRO AMBIENTE MANIPULANDO DADOS AOS QUAIS SE APLICA UMA POLÍTICA, SISTEMA QUE SUPORTA O PARTICIONAMENTO DE UMA APLICAÇÃO EM PELO MENOS UM PRIMEIRO OBJETO DE SOFTWARE E UM SEGUNDO OBJETO DE SOFTWARE

(51) Int.Cl.: G06F 21/53

(52) CPC: G06F 21/53

(30) Prioridade Unionista: 24/10/2003 US 10/693.749

(73) Titular(es): MICROSOFT TECHNOLOGY LICENSING, LLC

(72) Inventor(es): KENNETH D. RAY; MARCUS PEINADO; THEKKTHALACKAL VARUGIS KURIEN; PAUL ENGLAND

Relatório Descritivo da Patente de Invenção para **"SISTEMA QUE GERENCIA O PARTICIONAMENTO DE UMA APLICAÇÃO, MÉTODO DE UM PRIMEIRO OBJETO DE SOFTWARE QUE EXECUTA EM UM PRIMEIRO AMBIENTE MANIPULANDO DADOS AOS QUAIS SE APLICA UMA POLÍTICA, SISTEMA QUE SUPORTA O PARTICIONAMENTO DE UMA APLICAÇÃO EM PELO MENOS UM PRIMEIRO OBJETO DE SOFTWARE E UM SEGUNDO OBJETO DE SOFTWARE".**

CAMPO DA INVENÇÃO

[001] A presente invenção se refere geralmente ao campo de computação. Mais especificamente, a invenção provê um mecanismo que suporta o particionamento ou fatoração de aplicações de uma maneira que permite que operações que exigem uma medida de confiança ou segurança sejam integradas em software comum, não-seguro.

FUNDAMENTOS DA INVENÇÃO

[002] No campo de computação, há uma tensão entre, de um lado, sistemas que proporcionam um alto grau de segurança, e do outro lado, sistemas que proporcionam um grande número de recursos funcionais e um elevado grau de extensibilidade. Segurança no campo de computação depende da capacidade de se entender e prever o comportamento de um sistema de computação (isto é, o comportamento do software assim como do hardware) com um elevado grau de certeza - isto é, a capacidade de garantir que o sistema não irá, através de mau uso inadvertido ou ataque deliberado, se comportar de uma maneira diferente daquela para a qual ele foi projetado. Por exemplo, um sistema de computador que é projetado para proteger contra cópia, um material com direitos reservados, é digno de confiança apenas até o ponto em que podemos ter certeza de que o sistema na realidade realizará aquilo para o qual ele foi

projetado. Arquiteturas grandes, abertas, contudo, tendem a ser de difícil manejo e tendem a ser complexas, o que torna difícil analisar o seu comportamento, uma vez que há um grande número de variáveis que podem afetar esse comportamento. Atualmente, parece improvável que um programa complexo extenso tal como um sistema operacional de serviço completo, ou processador de texto, possa ter seu comportamento verificado com um alto grau de certeza. É possível gravar um pequeno programa cujo comportamento possa ser testado e verificado sob uma ampla variedade de condições e tipos de ataques, porém um tal programa seria capaz de realizar apenas um conjunto limitado de funções. Dessa forma, existe uma tensão entre prover um grande montante de funcionalidade e prover um elevado grau de segurança.

[003] Uma solução que foi proposta é a de executar dois sistemas lado a lado - um sistema grande que tem um alto grau de funcionalidade, e um outro sistema pequeno que tem um elevado grau de segurança. Dessa forma, um sistema operacional de serviço completo tal como WINDOWS XP poderia ser executado ao lado de um sistema operacional pequeno, de elevada segurança. Toda vez que ocorresse um evento no sistema operacional de serviço completo que precisasse ser realizado de uma maneira rigorosamente controlada, com um elevado grau de confiança, a tarefa poderia ser passada para o sistema operacional de alta garantia.

[004] Os sistemas operacionais proporcionam ambientes nos quais outros programas podem executar. Contudo, o simples fato de que dois sistemas operacionais podem existir lado a lado não resolve o problema de como uma determinada aplicação pode fazer uso de ambos os ambientes. Seria conveniente que uma aplicação utilizasse o ambiente de recursos completos para realizar a maioria das funções (isto é, aquelas que não exigem um alto grau de segurança), e se

utilizasse o ambiente de alta garantia para realizar funções que exigem um alto grau de segurança. Além disso, é conveniente utilizar esses dois ambientes de uma forma que proporcione uma experiência integrada de usuário.

[005] Devido ao anteriormente mencionado, há uma necessidade de um sistema que supere os empecilhos da técnica anterior.

SUMÁRIO DA INVENÇÃO

[006] A presente invenção provê um mecanismo pelo qual a funcionalidade de uma aplicação pode ser fatorada ou partilhada em múltiplas partes - isto é, aquelas ações que exigem um certo grau de segurança ou proteção, e aquelas que não exigem. De acordo com a invenção, uma aplicação é incorporada como pelo menos dois objetos de software: um objeto de software que opera em um ambiente de recursos plenos (porém de baixa garantia), e um outro objeto de software que opera em um ambiente de alta garantia (porém de recursos limitados). Quando o objeto no ambiente de recursos completos é operado, o mesmo pode encontrar dados que exigem certo grau de proteção (por exemplo, os dados podem exigir segredo, ou os mesmos podem ter que ser verificados no sentido de se determinar se os dados não foram violados). Quando o objeto de software que é operado no ambiente de recursos plenos encontra tais dados, esse objeto de software faz com que os dados sejam passados para o ambiente de alta garantia. O objeto de software operando no ambiente de alta garantia opera então sobre os dados. Qualquer entrada, saída, ou armazenamento dos dados que seja exigido enquanto os dados estão sendo processados é realizado utilizando-se o ambiente de alta garantia, para resistir à interceptação ou violação dos dados por eventos que surgem fora do ambiente de alta garantia.

[007] Os dois ambientes são hospedados por um componente base que proporciona a infra-estrutura (ou "encanamento") necessária

para que os dois ambientes se comuniquem. Por exemplo, um objeto de dados que precisa ser processado em um ambiente de alta garantia pode ter um invólucro gerado pelo componente base. Esse invólucro pode identificar o ambiente para o qual o objeto de dados deve ser encaminhado para processamento, e também pode prover um selo que permite que se verifique se o objeto de dados não foi modificado desde que foi envolvido pelo componente base. Dessa forma, um objeto de software pode passar o objeto de dados para o componente base, e o componente base será capaz de: (1) determinar para qual ambiente o objeto de dados deve ser passado, e (ii) verificar se o objeto de dados não foi violado desde que foi criado. Essa ação mencionada por último provê a infra-estrutura que permite que seja estabelecida a confiabilidade de um objeto através de plataformas: se um objeto é criado em uma primeira máquina (tendo um primeiro componente base), então o primeiro componente base assina o objeto como um atestado de que o objeto é o exato objeto que foi gerado em um ambiente de alta garantia conhecido daquele componente base. Quando o objeto é, então, aberto em uma outra máquina, a outra máquina pode verificar a assinatura e tomar uma decisão no sentido de se confia no signatário (Por exemplo, algumas máquinas e/ou componentes básicos podem ser melhores em garantir comportamento correto de seus ambientes hospedados do que outras máquinas; cada máquina pode decidir por si se confia na plataforma na qual um determinado objeto de dados foi criado).

[008] Outras características da invenção são descritas abaixo.

DESCRIÇÃO RESUMIDA DOS DESENHOS

[009] O resumo anterior, bem como a descrição detalhada que se segue das modalidades preferidas, é mais bem entendido quando lido em conjunto com os desenhos anexos. Com a finalidade de ilustrar a invenção, são mostradas nos desenhos construções exemplares da

invenção; contudo, a invenção não é limitada aos métodos e meios específicos revelados. Nos desenhos:

A Figura 1 é um diagrama de blocos de um ambiente de computação exemplar no qual podem ser implementados aspectos da invenção;

A Figura 2 é um diagrama de blocos de uma amplificação que é fatorada em funcionalidades constituintes;

A Figura 3 é um diagrama de blocos mostrando o roteamento de dados para componentes diferentes de uma aplicação;

A Figura 4 é um diagrama de blocos de um exemplo de uma interface de usuário para uma aplicação que pode ser fatorada;

A Figura 5 é um diagrama de blocos de uma arquitetura exemplar que suporta o uso de aplicações fatoradas;

A Figura 6 é um diagrama de blocos de uma hierarquia exemplar de ambientes nos quais uma aplicação fatorada pode ser usada;

A Figura 7 é um diagrama de blocos de um objeto de dados exemplar para uso em um ambiente que suporta o uso de uma aplicação fatorada;

A Figura 8 é um fluxograma de um processo exemplar através do qual os dados podem ser processados em uma aplicação fatorada.

DESCRIÇÃO DETALHADA DA INVENÇÃO

Visão Geral

[0010] A presente invenção provê um mecanismo que permite que uma aplicação seja partilhada ou "fatorada" em componentes seguros e em componentes não-seguros, e que permite que esses componentes operem juntos para prover uma experiência integrada de usuário com relação à aplicação. Por exemplo, um programa de processamento de texto poderia ser partilhado em um componente

não-seguro que realiza a maior parte da configuração, edição, impressão, correção ortográfica, correção gramatical, etc., funções que são associadas a um processador de texto, e um componente seguro que permite a exibição e edição de objetos de dados que precisam de alguma forma de proteção. O componente não-seguro poderia operar em um ambiente comum, aberto, tal como um sistema operacional comercial típico. O componente seguro poderia operar em um ambiente de alta garantia que permite que certos tipos de software operem com uma alta garantia de que o software se comportará de forma correta. A invenção provê vários recursos com relação a um tal cenário. Primeiramente, a invenção provê uma experiência integrada de usuário através dos dois componentes, de modo que, do ponto de vista do usuário, ele se sente, tanto quanto possível, como se estivesse utilizando uma única aplicação. Em segundo lugar, a invenção provê a infra-estrutura ou "encanamento" que permite que uma aplicação gerencie dados seguros assim como dados não-seguros, e que tais dados sejam utilizados através das partições.

[0011] Com relação à experiência de usuário, geralmente ocorre que o usuário começa utilizando a parte não-segura da aplicação. Os dados que surgem durante tal uso (por exemplo, itens confidenciais de texto, em um documento de processamento de texto, números financeiros confidenciais em uma planilha, etc.), são preferivelmente integrados na experiência de usuário de certo modo pela parte não-segura da aplicação, embora os dados não possam ser de fato exibidos pela parte não-segura. Por exemplo, se há um item de texto secreto ou confidencial em um documento de processamento de texto, a parte não-segura do processador de texto pode ser capaz de exibir uma caixa que representa o item, junto com algum tipo de gráfico (por exemplo, linhas ilegíveis indicativas de texto) que representa o fato de que o texto existe embora não possa ser exibido. Em um exemplo, o

usuário poderia clicar na caixa ou gráfico, e a parte segura da aplicação seria então ativada para exibir o texto no mesmo local na tela onde aparece a caixa. Dessa forma, as partes seguras e não-seguras da aplicação são integradas com relação à experiência de usuário.

[0012] Com relação à infra-estrutura, a invenção provê mecanismos que permitem que objetos de dados em um ambiente sejam encaminhados para um outro ambiente. Tipicamente, um objeto de dados que é secreto ou confidencial - e dessa forma precisa ser processado pela parte segura - será codificado de modo que a parte não-segura não pode ler o mesmo. Dessa forma, um tal objeto será tipicamente envolvido em uma série de invólucros por cada componente que desempenhou uma parte na geração do objeto de dados. Cada invólucro contém preferivelmente um identificador do componente que anexou o invólucro, bem como um selo que permite que seja verificada a integridade dos dados dentro do invólucro. O invólucro externo é preferivelmente anexado pela fonte de confiança na máquina que criou o invólucro - isto é, um componente base que hospeda os vários ambientes em uma única máquina, e que foi verificado por alguma autoridade conhecida como sendo digna de confiança em seu comportamento. Esse invólucro externo permite que o componente base sirva como um encaminhador. Dessa forma, quando a parte não-segura encontra o objeto, ela pode não ser capaz de ler o objeto, porém, pode identificar o objeto como um objeto que precisa ser enviado para um outro ambiente para processamento. Dessa forma, a parte não-segura envia o objeto para o componente base que, então, encaminha o objeto para o ambiente correto com base em qual ambiente esteja identificado no invólucro.

[0013] Adicionalmente, se o objeto é criado em uma primeira máquina e aberto em uma segunda máquina, a segunda máquina

pode determinar a confiabilidade do objeto mediante verificação da assinatura que é parte do invólucro externo. Deve-se entender que quando o componente base envolve e sela um objeto, o componente base está essencialmente atestando que, durante a criação do objeto, o componente base realizou corretamente suas funções em termos de proteger o processo de criação contra violação externa. (O componente base provê, tipicamente, os mecanismos que permitem que um ambiente de alta garantia se proteja contra violação - por exemplo, um módulo de processador de confiança (TPM), um mecanismo de isolamento de memória, etc.). Alguns componentes básicos podem realizar essa função melhor do que outros, de modo que qualquer máquina na qual o objeto de dados é aberto pode tomar uma decisão sobre se confia em que o objeto foi na realidade criado de acordo com as exigências de segurança que se aplicam àquele objeto. Por exemplo, se o objeto constitui texto que foi introduzido pelo teclado, o ambiente de alta garantia pode receber entrada a partir do teclado de uma forma segura, porém a capacidade do ambiente de receber entrada de teclado de forma segura pode depender da capacidade do componente base em proteger o caminho a partir do teclado até o ambiente de alta garantia. Uma vez que o invólucro contém um selo, ou assinatura, gerado por um componente base específico, o invólucro suporta o modelo de confiança no qual máquinas diferentes podem tomar decisões sobre quão digno de confiança é um objeto de dados com base na segurança presumida da máquina na qual o objeto de dados foi criado.

[0014] O que se segue descreve sistemas, métodos, e mecanismos que suportam o uso de aplicações fatoradas ou partilhadas.

Arranjo de Computação Exemplar

[0015] A Figura 1 mostra um ambiente de computação exemplar

no qual aspectos da invenção podem ser implementados. O ambiente 100 de sistema de computação é apenas um exemplo de um ambiente de computação adequado e não serve para sugerir qualquer limitação ao escopo de uso ou funcionalidade da invenção. Nem tampouco deve o ambiente 100 de computação ser interpretado como tendo qualquer dependência ou exigência relacionada a qualquer um componente ou combinação de componentes ilustrados no ambiente 100 de operação exemplar.

[0016] A invenção é operacional com vários outros ambientes ou configurações de sistema de computação de uso geral ou de uso especial. Exemplos de sistemas, ambientes, e/ou configurações de computação, conhecidos, que podem ser adequados para uso com a presente invenção incluem, porém, que não são limitados a computadores pessoais, computadores de servidor, dispositivos portáteis ou laptop, sistemas de múltiplos processadores, sistemas baseados em microprocessador, aparelhos conversores de sinal de freqüência, meios eletrônicos programáveis de consumidor, PCs de rede, minicomputadores, computadores de grande porte, sistemas integrados, ambientes de computação distribuídos que incluem quaisquer dos sistemas ou dispositivos acima, e semelhantes.

[0017] A invenção pode ser descrita no contexto geral de instruções executáveis por computador, tais como módulos de programa, sendo executados por um computador. Geralmente, os módulos de programa incluem rotinas, programas, objetos, componentes, estruturas de dados, etc. que realizam tarefas específicas ou implementam tipos de dados abstratos específicos. A invenção também pode ser praticada em ambientes de computação distribuídos onde as tarefas são realizadas por dispositivos remotos de processamento que são ligados através de uma rede de comunicação ou outro meio de transmissão de dados. Em um ambiente de

processamento distribuído, os módulos de programa e outros dados podem estar localizados em uma mídia de armazenamento de computador local assim como em uma mídia de armazenamento de computador remoto incluindo dispositivos de armazenamento de memória.

[0018] Com referência à Figura 1, um sistema exemplar para implementar a invenção inclui um dispositivo de computação de uso geral na forma de um computador 110. Componentes do computador 110 podem incluir, porém não são limitados a, uma unidade 120 de processamento, uma memória 130 de sistema, e um barramento 121 de sistema que acopla vários componentes de sistema, incluindo a memória de sistema, à unidade 120 de processamento. A unidade 120 de processamento pode representar múltiplas unidades lógicas de processamento tais como aquelas suportadas em um processador de múltiplos encadeamentos. O barramento 121 de sistema pode ser qualquer um dos vários tipos de estruturas de barramento incluindo um barramento de memória ou controladora de memória, um barramento periférico, e um barramento local utilizando qualquer uma de uma variedade de arquiteturas de barramento. Como exemplo, e não limitação, tais arquiteturas incluem barramento de Arquitetura Padrão da Indústria (ISA), um barramento de Arquitetura de Microcanal (MCA), um barramento ISA Otimizado (EISA), um barramento local da Associação de Padrões Eletrônicos de Vídeo (VESA), e um barramento de Interconexões de Componentes Periféricos (PCI) (também conhecido como barramento Mezzanine). O barramento 121 de sistema também pode ser implementado como uma conexão de ponto a ponto, estrutura de comutação, ou semelhante, entre os dispositivos de comunicação.

[0019] O computador 110 inclui tipicamente uma variedade de mídias legíveis por computador. Mídias legíveis por computador

podem ser quaisquer mídias disponíveis que possam ser acessadas pelo computador 110 e incluem mídias voláteis e não-voláteis; mídias removíveis e não-removíveis. Como exemplo, e não como limitação, mídia legível por computador pode compreender mídia de armazenamento de computador e mídia de comunicação. Mídia de armazenamento de computador inclui mídia volátil e não-volátil, removível e não-removível implementada em qualquer método ou tecnologia para armazenamento de informação tal como instruções legíveis por computador, estruturas de dados, módulos de programa ou outros dados. Mídia de armazenamento de computador inclui, porém, não é limitada a, RAM, ROM, EEPROM, memória flash ou outra tecnologia de memória, CD-ROM, discos versáteis digitais (DVD) ou outro armazenamento de disco ótico, cassetes magnéticos, fita magnética, armazenamento de disco magnético ou outros dispositivos magnéticos de armazenamento, ou qualquer outro meio que possa ser usado para armazenar a informação desejada e que possa ser acessado pelo computador 110. Mídia de comunicação incorpora tipicamente instruções legíveis por computador, estruturas de dados, módulos de programa e outros dados em um sinal modulado de dados tal como uma onda portadora ou outro mecanismo de transporte que inclui qualquer mídia de distribuição de informação. O termo "sinal modulado de dados" significa um sinal que tem uma ou mais de suas características determinada ou alterada de tal modo a codificar informação no sinal. Como exemplo, e não como limitação, mídia de comunicação inclui mídia de ligação física tal como uma rede de ligação física ou conexão física direta, e mídia sem fio tal como mídia acústica, de RF, infravermelha e outra mídia sem fio. Combinações de quaisquer dos mencionados acima também devem ser incluídas no escopo de mídia legível por computador.

[0020] A memória 130 de sistema inclui mídia de armazenamento

de computador na forma de memória volátil e/ou não-volátil tal como memória de leitura (ROM) 131 e memória de acesso aleatório (RAM) 132. Um sistema base de entrada/saída (BIOS) 133, contendo as rotinas básicas que ajudam a transferir informação entre elementos dentro do computador 110, tal como durante a inicialização, é tipicamente armazenado na ROM 131. RAM 132 contém tipicamente dados e/ou módulos de programa que são imediatamente acessíveis à, e/ou presentemente sendo operados pela, unidade de processamento 120. Como exemplo, e não como limitação, a Figura 1 ilustra o sistema operacional 134, programas 135 de aplicação, outros módulos 136 de programa e dados 137 de programa.

[0021] O computador 110 também pode incluir outra mídia de armazenamento de computador removível/não-removível, volátil/não-volátil. Apenas como exemplo, a Figura 1 ilustra uma unidade 140 de disco rígido que lê a partir de, ou grava em uma mídia magnética não-removível, não-volátil, uma unidade 151 de disco magnético que lê a partir de, ou grava em um disco 152 magnético removível, não-volátil, e uma unidade 155 de disco ótico que lê a partir de, ou grava em um disco 156 ótico removível, não-volátil, tal como um CD-ROM ou outra mídia ótica. Outra mídia de armazenamento de computador removível/não-removível, volátil/não-volátil que pode ser usada no ambiente de operação exemplar, inclui, porém, não é limitada a; cassetes de fita magnética; placas de memória flash; discos versáteis digitais; fita de vídeo digital, RAM de estado sólido, ROM de estado sólido, e semelhantes. A unidade 141 de disco rígido é conectada tipicamente ao barramento 121 de sistema através de uma interface de memória não-removível tal como a interface 140, e unidade 151 de disco magnético e unidade 155 de disco ótico são conectadas tipicamente ao barramento 121 de sistema através de uma interface de memória removível, tal como a interface 150.

[0022] As unidades e suas mídias de armazenamento de computador, associadas, discutidas acima e ilustradas na Figura 1, proporcionam armazenamento de instruções legíveis por computador, estruturas de dados, módulos de programa e outros dados para o computador 110. Na Figura 1, por exemplo, a unidade 141 de disco rígido é ilustrada como armazenando o sistema operacional 144, programas 145 de aplicação, outros módulos 146 de programa, e dados 147 de programa. Observar que esses componentes podem ser idênticos ou diferentes do sistema operacional 134, programas 135 de aplicação, outros módulos 136 de programa, e dados 137 de programa. O sistema operacional 144, programas 145 de aplicação, outros módulos 146 de programa, e dados 147 de programa recebem aqui números diferentes para ilustrar que, no mínimo, eles são cópias diferentes. Um usuário pode introduzir comandos e informação no computador 20 através de dispositivos de entrada, tal como um teclado 162 e dispositivo indicador 161, comumente referido como um mouse, trackball, ou mesa sensível ao toque. Outros dispositivos de entrada (não mostrados) podem incluir um microfone, joystick, mesa de jogos, antena de prato de satélite, scanner, ou semelhantes. Esses e outros dispositivos de entrada são freqüentemente conectados à unidade 120 de processamento através de uma interface 160 de entrada de usuário que é acoplada ao barramento de sistema, mas pode ser conectada mediante outra interface e estruturas de barramento, tal como uma porta paralela, porta de jogos ou um barramento serial universal (USB). Um monitor 191 ou outro tipo de dispositivo de vídeo também é conectado ao barramento 121 de sistema através de uma interface, tal como uma interface 190 de vídeo. Além do monitor, os computadores também podem incluir outros dispositivos periféricos de saída tais como alto-falantes 197 e impressora 196, que podem ser conectados através de uma interface

195 periférica de saída.

[0023] O computador 110 pode operar em um ambiente de rede utilizando conexões lógicas para um ou mais computadores remotos, tal como um computador remoto 180. O computador remoto 180 pode ser um computador pessoal, um servidor, um encaminhador, um PC de rede, um dispositivo não hierárquico ou outro nó de rede comum e, tipicamente, inclui muitos/todos os elementos descritos acima em relação ao computador 110, embora apenas um dispositivo 181 de armazenamento de memória tenha sido ilustrado na Figura 1. As conexões lógicas ilustradas na Figura 1 incluem uma rede de área local (LAN) 171 e uma rede remota (WAN) 173, mas pode incluir também outras redes. Tais ambientes de rede são comuns em escritórios, redes de computadores empresariais, intranets e na Internet.

[0024] Quando usado em um ambiente de rede LAN, o computador 110 é conectado a LAN 171 através de uma interface de rede ou adaptador 170. Quando usado em um ambiente de rede WAN, o computador 110 inclui tipicamente um modem 172 ou outro meio para estabelecer comunicação através da WAN 173, tal como a Internet. O modem 172, o qual pode ser interno ou externo, pode ser conectado ao barramento 121 de sistema através da interface 160 de entrada de usuário, ou outro mecanismo apropriado. Em um ambiente de rede, módulos de programa ilustrados em relação ao computador 110, ou partes do mesmo, podem ser armazenados no dispositivo remoto de armazenamento de memória. Como exemplo, e não como limitação, a Figura 1 ilustra programas 185 de aplicação remota como residindo no dispositivo 181 de memória. Será considerado que as conexões de rede mostradas, são exemplares, e que podem ser usados outros meios de estabelecer uma ligação de comunicação entre os computadores.

Aplicação Partilhada ou "Fatorada"

[0025] Software, tal como um programa de aplicação, realiza tipicamente várias funções diferentes e opera sobre vários tipos diferentes de dados. Neste sentido, uma aplicação pode ser vista como uma compilação de diferentes funcionalidades. Pode ser útil decompor uma aplicação em suas várias funcionalidades diferentes, de modo que essas funcionalidades diferentes possam ser realizadas separadamente (por exemplo, atribuídas a ambientes proporcionando níveis diferentes de segurança). A Figura 2 mostra como uma aplicação pode ser decomposta em suas várias funcionalidades diferentes.

[0026] A aplicação 135 compreende várias funcionalidades diferentes 202(1), 202(2),..., 202(n), 202(n+1), 202(n+2),..., 202(n+m). Por exemplo, a aplicação 135 pode ser um programa de processamento de texto, e as funcionalidades separadas podem ser editar, ver, imprimir, monitorar mudanças, etc. Deve ser observado que, embora a Figura 2 mostre a aplicação 135 como tendo n+m funcionalidades distintas, há um certo critério em se decidir o que é uma funcionalidade distinta. Por exemplo, todas as operações de imprimir podem ser vistas como uma única funcionalidade, ou então a operação de imprimir pode ser vista como compreendendo duas ou mais funcionalidades diferentes (por exemplo, imprimir uma página versus imprimir o documento inteiro). Adicionalmente, como discutido adicionalmente abaixo, a mesma operação básica pode ser vista como múltiplas funcionalidades dependendo de sobre qual tipo de dados se está operando. Por exemplo, a operação básica de visualizar um documento pode ser considerada como sendo uma de duas funcionalidades diferentes, dependendo de se os dados sendo vistos são seguros ou não-seguros. Dessa forma, poderia haver duas funcionalidades separadas de visualização, uma que é segura em

certo aspecto, e outra que não é segura. Essencialmente, decompor um programa em suas funcionalidades constituintes é uma questão de traçar limites em torno das várias tarefas que o programa realiza (e/ou dos vários tipos diferentes de dados sobre o qual o programa opera), e, em geral não há exigência específica em relação a como esses limites são traçados.

[0027] Em um exemplo, as funcionalidades podem ser agrupadas de tal modo que as funcionalidades 202(1) até 202(n) são parte de uma primeira "partição" 206(1), e as funcionalidades 202(n+1) até 202(n+m) são parte de uma segunda partição 206(2). Pode ser conveniente agrupar as funcionalidades desta maneira, de modo que funcionalidades na mesma partição podem ser tratadas similarmente. Por exemplo, a partição 206(1) pode incluir as funcionalidades da aplicação 135 que envolvem dados comuns, não-seguros, enquanto que a partição 206(2) pode incluir as funcionalidades da aplicação 135 que envolvem dados secretos ou seguros (ou dados que de outra forma exigem um certo nível de proteção). Dessa forma, funcionalidades na partição 206(1) podem ser realizadas em ambiente aberto comum (por exemplo, o ambiente provido por um sistema operacional comercial comum), enquanto que as funcionalidades na partição 206(2) podem ser executadas em um ambiente de alta garantia. (Ambientes de alta garantia são discutidos mais particularmente abaixo).

[0028] A Figura 3 mostra um exemplo de como as partições podem ser usadas para permitir que uma única aplicação 135 gerencie dados seguros assim como dados não-seguros. No exemplo da Figura 3, a aplicação 135 realiza operações 302 que exigem um alto grau de proteção (por exemplo, operações envolvendo dados secretos), e também realiza operações 304 que exigem um grau inferior de proteção ou nenhuma proteção (por exemplo, operações que não

envolvem dados secretos). ("Operações" sobre dados, neste exemplo, podem incluir qualquer tipo de manipulação de dados, tal como realizar entrada ou saída dos dados, realizar um cálculo sobre os dados, etc.). Neste exemplo, a funcionalidade que não envolve operar sobre dados secretos é gerenciada pela partição 206(1) da aplicação 135, e a funcionalidade que envolve operação sobre dados secretos (ou dados que de outra forma exigem algum tipo de proteção) é gerenciada pela partição 206(2). Por exemplo, se a aplicação 135 é um programa de processamento de texto, então exibir um documento comum (não protegido) pode ser realizado pela partição 206(1), e exibir um documento secreto pode ser realizado pela partição 206(2). Como um outro exemplo, se a aplicação 135 é um programa de comercialização de ações, a partição 206(1) pode conter funcionalidade que permite que um usuário consulte o valor atual de uma ação, enquanto a partição 206(2) pode permitir que o usuário compre e venda ações do capital após autenticar o usuário (neste caso, as credenciais de autenticação do usuário, e os detalhes de suas contas financeiras, são um tipo de dados secretos).

[0029] Deve ser entendido que a Figura 3 demonstra que partições diferentes de uma aplicação podem ser usadas para lidar com dados secretos e não secretos, porém, que a Figura 3 não é limitada a qualquer mecanismo específico para implementar ou usar uma aplicação partilhada. Uma arquitetura exemplar é discutida abaixo, em conexão com as Figuras 4-8, a qual suporta a partilha de uma aplicação, e o uso das partições de uma forma que proporciona uma experiência integrada de usuário.

Aplicação Partilhada Exemplar

[0030] Como discutido acima, as várias funcionalidades de uma aplicação podem ser partilhadas de acordo com qualquer critério. No exemplo da Figura 3, a funcionalidade de uma aplicação é partilhada

de acordo com o fato de se a aplicação está manipulando dados seguros ou com dados não-seguros. Quando uma aplicação é partilhada, é conveniente integrar as várias partições em uma única experiência de usuário. A Figura 4 mostra uma interface de usuário exemplar para uma aplicação partilhada. (Deve ser entendido que dados "seguros" se refere aos dados que exigem um certo tipo de proteção, embora a invenção não seja limitada a qualquer tipo específico de proteção. Por exemplo, os dados podem exigir proteção no sentido de "segredo", em cujo caso os dados podem ser codificados. Como um outro exemplo, os dados podem exigir proteção no sentido de serem verificáveis - isto é, a capacidade de se determinar se os dados não foram modificados desde um momento de referência - em cujo caso os dados podem ser assinados).

[0031] A interface 400 de usuário é uma interface para um programa de processamento de texto. (Deve ser entendido que um programa de processamento de texto é um exemplo conveniente de uma aplicação, embora o mesmo de forma alguma seja o único exemplo). No exemplo da Figura 4, a interface 400 de usuário exibe diversos itens 402(1) até 402(5) que são parte de um documento de processamento de texto. O item 402(1) é texto comum não-seguro. O item 402(2) é um gráfico 404 não-seguro. Os itens 402(3), 402(4), e 402(5) são itens de texto seguro. A partição não-segura da aplicação realiza todo processamento que não envolve o gerenciamento de dados seguros. Neste exemplo, esse processamento inclui a exibição (e possivelmente a ação de editar, imprimir, salvar, etc.) de itens não-seguros 402(1) e 402(2), bem como a configuração de todos os itens (até mesmo os itens seguros). A partição não-segura é capaz de configurar itens seguros porque se supõe que embora o conteúdo dos itens 402(3), 402(4), e 402(5) possam ser secretos, a exigência desses itens não é secreta. Dessa forma, a partição não-segura é capaz de

exibir itens 402(3) até 402(5) como traços ilegíveis indecifráveis 405, dessa forma proporcionando ao usuário pelo menos alguma experiência de usuário com relação ao conteúdo que a partição não-segura não é capaz de processar.

[0032] Se o usuário pretender ver os itens 402(3), 402(4) e 402(5) de conteúdo, o usuário pode, por exemplo, clicar nos traços ilegíveis 405 que representam um determinado item de conteúdo. Essa ação pode fazer com que a partição segura seja ativada. O conteúdo contido em um dos itens seguros pode, então, ser passado para a partição segura para gerenciamento. Em um exemplo, a partição segura pode então exibir o item de conteúdo efetivo em uma janela que é sobreposta ao local para o conteúdo que foi configurado pela partição não-segura - por exemplo, quando a partição segura exibe item 402(3) de conteúdo, ela pode fazer isso mediante colocação de uma imagem daquele item de conteúdo em cima da região onde os traços ilegíveis 405 que representam o item 402(3) de conteúdo foram previamente mostrados. Embora o cenário descrito acima envolva a interação do usuário com duas peças diferentes de software (isto é, a partição segura e a partição não-segura), pode parecer ao usuário que ele está interagindo com uma única aplicação, desse modo proporcionando uma experiência integrada de usuário através das duas partições.

[0033] A Figura 4 mostra um exemplo no qual a aplicação partilhada é um programa de processamento de texto. Outros exemplos incluem:

Uma planilha, onde o usuário pode introduzir fórmulas e elaborar tabelas, etc., na partição não-segura, porém, onde os dados efetivos a serem trabalhados estão disponíveis apenas na partição segura. Dessa forma, a partição não-segura exibe alguns dados portadores de posição (por exemplo, "xxxx") em cada célula. A avaliação e a

renderização dos dados na célula são realizadas pela partição segura da aplicação. Quando o usuário aperta o botão de "calcular", aparece uma janela (gerada pela partição segura) que exibe os valores de célula nas fileiras e colunas apropriadas, que são calculadas com base nos dados subjacentes que estão disponíveis para a partição segura. Nessa implementação, a partição segura terá um mecanismo de avaliação e uma rotina de renderização (trivial), mas não precisará incorporar outros aspectos da interface de usuário da planilha, janelas de ajuda, elaboradores de fórmula, etc.

[0034] Uma aplicação de comercialização de apólices, onde são atribuídas funções às partições segura e não-segura conformemente. Por exemplo, exibir gráficos de comercialização e informação de ações (que estão publicamente disponíveis) pode ser realizado pela partição não-segura. A partição segura, por outro lado, permite que o usuário introduza o símbolo de comercialização de ações, o preço e número de cotas a serem compradas e vendidas pelo usuário, e envie essa informação para um servidor remoto de comercialização, após primeiramente verificar a identidade do servidor.

[0035] Um processador de texto que emite um documento em um formato de imagem (por exemplo, Formato de Documento Portátil, ou "PDF"). A partição segura é capaz de ler e exibir a imagem. Essa renderização dos documentos é chamada de "fac-símile" do documento. O documento de processamento de texto subjacente (isto é, a versão do documento que contém códigos de formatação e caracteres de texto, em vez de apenas uma imagem) e seu fac-símile são enviados pela partição não-segura para a partição segura, e a partição segura exibe o fac-símile. O usuário assina o fac-símile (ou um sumário do documento e o fac-símile) utilizando um agente de assinatura seguro em um ambiente seguro. O fac-símile assinado (chamado de "fac-símile-0") e o documento subjacente são retornados

em uma gota para a partição não-segura. Quando um verificador recebe a gota contendo o documento, fac-símile-0 e a assinatura do fac-símile, o verificador primeiramente renderiza uma nova cópia do fac-símile (chamada de fac-símile-1) com base no documento de processamento de texto subjacente. O verificador transmite o fac-símile-0; o fac-símile-1; o documento do Word; e a assinatura através da partição segura a qual verifica se os fac-símiles são idênticos, e se a assinatura no fac-símile-0 é válida. Embora esse modelo não proporcione segredo do documento (uma vez que o fac-símile-1 do documento ainda pode ser renderizado pela partição não-segura), o mesmo serve para verificar a integridade do documento - isto é, o fato de se o documento é aquele que foi originalmente criado, e não foi modificado.

[0036] A Figura 5 mostra uma arquitetura que suporta a partilha de uma aplicação. Na Figura 5, há dois ambientes 502(1) e 502(2) nos quais o software pode ser executado. Neste exemplo, há quatro processadores separados 504(1), 504(2), 504(3), e 504(4). Neste contexto, "processador" não se refere a um microprocessador, porém mais propriamente a um componente de software que de alguma forma processa os dados para a aplicação. Os processadores 504(1) e 504(2) operam no ambiente 502(1), e os processadores 504(3) e 504(4) operam no ambiente 502(2). Por exemplo, os processadores 504(1) e 504(3) podem ser a partição não-segura e a partição segura, respectivamente, de uma única aplicação de processamento de texto. Um componente base 508 hospeda os ambientes 502(1) e 502(2), de modo que os dois ambientes podem coexistir em uma única máquina. A invenção não é limitada a qualquer tipo específico de componente base 508; alguns exemplos de componente base 508 são um monitor de máquina virtual (VMM), um núcleo externo, um micronúcleo, ou um hipervisor. Por exemplo, os ambientes 502(1) e 502(2) podem ser

sistemas operacionais separados hospedados por um VMM ou por um hipervisor. Como um outro exemplo, o componente base 508 poderia ser um dos sistemas operacionais - por exemplo, um sistema operacional de alta garantia que proporciona alguma funcionalidade de usuário, e também impõe a separação entre ele próprio e o outro sistema operacional (de alta garantia).

[0037] Um componente que executa como parte do componente base 508 é um monitor 510 de referência. O monitor 510 de referência realiza a função de rotear um determinado objeto de dados para o ambiente correto, para processamento. Por exemplo, os dados 506 rotulados podem ser encontrados (ou gerados ou introduzidos) durante a execução de uma aplicação, e o monitor 510 de referência faz com que os dados 506 rotulados sejam encaminhados para o ambiente correto. Os dados 506 rotulados são associados a uma marca de identificação que permite que o monitor 510 de referência determine para qual ambiente os dados rotulados devem ser encaminhados. Adicionalmente, os dados 506 rotulados também podem conter marcas adicionais que permitem que o ambiente de destino determine para qual processador os dados devem ser fornecidos para processamento. Uma estrutura exemplar de dados 506 rotulados é discutida abaixo em conexão com a Figura 7.

[0038] Dados 506 rotulados podem ser encontrados (ou gerados ou introduzidos) em qualquer local, embora mais tipicamente os dados rotulados serão encontrados em um ambiente e serão passados para um outro ambiente. Por exemplo, a parte não-segura de uma aplicação de processamento de texto pode encontrar dados rotulados que contém algum texto secreto. A aplicação de processamento de texto pode, então, exibir uma representação do texto da forma mostrada na Figura 4 (por exemplo, como traços indecifráveis). Se o usuário pretender exibir o texto real (por exemplo, mediante ação de

clicar a área reservada para os dados), a partição não-segura da aplicação (por exemplo, processador 504(1) operando no ambiente 502(1)) pode fornecer os dados 506 rotulados ao monitor 510 de referência, o qual fornecerá então os dados ao ambiente 502(2). O ambiente 502(2) pode, então, encaminhar os dados para o processador correto (por exemplo, processador 504(3)), o qual pode então exibir os dados no local apropriado na tela da forma descrita acima em conexão com a Figura 4.

[0039] Deve ser entendido que, dentro do modelo da Figura 5, as aplicações são feitas essencialmente de processadores - isto é, os componentes que podem lidar com tipos especificados diferentes de dados ou realizar certas categorias de tarefas - e partições diferentes de uma aplicação utilizam processadores diferentes. Por exemplo, os processadores 504(1) e 504(3) podem representar os processadores seguro e não-seguro de uma única aplicação (por exemplo, uma aplicação de processamento de texto), onde os dados são encaminhados para qualquer um dos processadores 504(1) ou 504(3) dependendo de se os dados são dados seguros ou não-seguros.

[0040] Deve ser entendido que não há exigência em relação a quais tipos de ambientes são disponíveis, mas pode ser particularmente útil para um ambiente ser um sistema operacional de alta garantia, e para o outro ambiente ser um sistema operacional aberto, de serviço completo, comum. Um sistema operacional de "alta garantia" é aquele que proporciona um nível relativamente elevado de garantia de que irá realizar suas funções esperadas de forma correta. Devido ao fato de que todo software (incluindo um sistema operacional) é associado a uma especificação que descreve a função esperada do software, e devido ao fato de que todo software está sujeito a algum nível de falhas ou ataques que fazem com que o software se comporte de uma forma inesperada, um sistema

operacional de "alta garantia" é aquele que provê um nível relativamente alto de confiança de que o sistema operacional se comportará de acordo com sua especificação. (A maioria dos softwares comerciais vem com uma especificação escrita, explícita, embora uma especificação nesse contexto também possa constituir um entendimento não escrito, implícito de como o software deve se comportar). Alta garantia não é a mesma coisa que segurança, embora alta garantia possa ser usada para se obter segurança. Por exemplo, um processador para dados secretos pode ser projetado para operar no sistema operacional de alta garantia; até o ponto em que a capacidade do processador em proteger os dados secretos depende do comportamento correto do ambiente no qual ele opera (por exemplo, a execução correta de chamadas de sistema, correto isolamento de processo, etc.), o processador pode prover um nível de segurança que poderia não ser conseguido se o processador tivesse que operar em um sistema operacional de serviço completo, comum. (A contrapartida para alta garantia é que um ambiente de alta garantia pode ter um escopo de funcionalidade muito limitado; quanto maior for o programa, mais difícil é verificar se o programa se comportará corretamente em uma ampla variedade de circunstâncias. Dessa forma, um sistema operacional comercial de serviço completo tal como WINDOWS XP tipicamente não seria considerado de alta garantia).

[0041] No contexto da descrição acima de alta garantia, pode ser considerado que freqüentemente é útil partilhar uma aplicação em um processador de baixa segurança que lida com dados que não exigem segurança significativa, e um processador de alta segurança que lida com dados que exigem mais segurança. O processador de alta segurança pode operar em um ambiente de alta garantia, e o processador de baixa segurança pode operar em um ambiente de baixa garantia.

Arquitetura Exemplar para Suportar uma Aplicação Partilhada

[0042] A Figura 6 mostra uma arquitetura exemplar 600 na qual uma aplicação partilhada pode operar. A arquitetura 600 inclui um componente base 508, cuja função é a de hospedar os vários ambientes nos quais irão operar as partições de uma aplicação. O componente base 508, como assinalado acima, pode assumir várias formas, tal como um VMM, um núcleo externo, um micronúcleo, um hipervisor, etc. O componente base 508 tem a funcionalidade de hospedar vários ambientes (por exemplo, sistemas operacionais), e também gerencia (e limita) a interação entre esses ambientes. A hospedagem de vários ambientes pode ser realizada utilizando-se várias técnicas. Por exemplo, o componente base 506 pode expor, a cada um dos vários sistemas operacionais, uma "máquina virtual" independente; o sistema operacional controla, então, o "hardware virtual" de máquina virtual; e o componente base 506 emite instruções para o hardware "real" as quais se baseiam (mas não são necessariamente idênticas) às instruções que os sistemas operacionais forneceram às máquinas virtuais. (Em termos gerais, essa é a forma como funciona o VMM tradicional). Como um outro exemplo, o componente base 506 pode atribuir certos dispositivos, e certos segmentos do espaço de endereço físico da máquina, aos sistemas operacionais diferentes, e pode impor a atribuição mediante ação de permitir que cada sistema operacional controle apenas os dispositivos a ele atribuídos e a parte do espaço de endereço a ele atribuída. A invenção não é limitada a qualquer modalidade específica de um componente base; supõe-se apenas, com o propósito da Figura 6, que há um componente base que é capaz de hospedar vários ambientes de modo que esses vários ambientes podem coexistir em uma única máquina com um certo grau de isolamento mútuo.

[0043] No exemplo da Figura 6, o componente base 506 hospeda

os sistemas operacionais 602(1) até 602(4). O sistema operacional 602(1) é uma instância do sistema operacional WINDOWS XP, ou de um outro sistema operacional de uso geral; o sistema operacional 602(2) é uma instância do sistema operacional Linux; o sistema operacional 603(3) é um "nexus" - isto é, o sistema operacional de alta garantia (dentro do significado descrito acima), que pode prover funcionalidade limitada, porém, uma alta garantia de que essa funcionalidade será realizada de forma correta; sistema operacional 602(4) pode ser um outro sistema operacional de uso geral, tal como OS/2. Em geral, o componente base 508 pode hospedar um número arbitrário de sistemas operacionais (ou outros tipos de ambientes), e os quatro sistemas operacionais mostrados na Figura 6 são simplesmente um exemplo.

[0044] Dentro de um determinado sistema operacional, é possível operar um exemplar de software de nível de aplicação. Por exemplo, o programa (604) de processamento de texto MICROSOFT WORD e o programa (606) de planilha MICROSOFT EXCEL operam sob o sistema 602(1) operacional WINDOWS XP. O sistema 602(2) operacional Linux, nexus 602(3), e o sistema 602(4) operacional OS/2 também podem executar seus próprios programas de aplicação. Neste exemplo, os programas denominados "Word-let" (608) e "Excel-let" (610) são executados sob nexus 602(3). "Word-let" 608 é o programa seguro que opera com o WORD 604 quando o WORD precisa lidar com dados seguros. Similarmente, "Excel-let" 610 gerencia dados seguros para EXCEL 606. Dessa forma, no exemplo da Figura 4, o WORD 604 pode ser o programa (ou "processador") que gerencia itens de dados não-seguros, configura itens de dados dentro de uma janela, e exibe os traços indecifráveis que representam itens seguros. Word-let 608 pode ser o programa que abre um item seguro e renderiza o item na região da janela que o WORD 604 reserva para o

item. Essencialmente, o WORD 604 e Word-let 608 representam, em conjunto, uma partição (ou "fatoração") da funcionalidade de uma aplicação de processamento de texto através de diferentes processadores que operam em ambientes diferentes. Similarmente, EXCEL 606 e Excel-let 610 podem ter uma relação similar, isto é, EXCEL 606 lida com a maior parte das funções de planilha que não envolvem dados seguros, e Excel-let 610 lida com os dados seguros em nome do EXCEL.

[0045] Como discutido acima, nexus 602(3) é um sistema operacional de alta garantia que pode prover um ambiente no qual uma aplicação de confiança (isto é, uma aplicação cujo comportamento é suficientemente garantido de que se pode confiar a ela operações confidenciais tais como a abertura de dados secretos) pode operar. Contudo, nexus 602(3) provavelmente proporcionará funcionalidade muito limitada. Dessa forma, é útil partilhar uma aplicação de processamento de texto, por exemplo, em WORD 604 e Word-let 608 de modo que o WORD possa usar os vastos recursos disponíveis em um sistema operacional de uso geral para prover ampla funcionalidade, e o Word-let pode usar a natureza de alta garantia do ambiente, provida pelo nexus 602(3), para realizar um conjunto (provavelmente limitado) de funções sensíveis com um alto grau de confiabilidade.

[0046] A Figura 6 mostra um componente base que proporciona a infra-estrutura para suportar a partilha de aplicações. O que se segue é uma descrição de alguns dos recursos que a infra-estrutura pode incluir:

[0047] Serviço de registro e diretório: O componente base pode prover uma interface através da qual os ambientes hospedados pelo componente base podem se registrar junto ao componente base. O componente base também pode prover métodos através dos quais os

ambientes hospedados podem ser inicializados (ou o componente base pode ser um dos ambientes hospedados). O nível de garantia de um ambiente é meta-informação associada ao ambiente, e o componente base pode prover um serviço opcional para acessar e consultar essa informação de uma forma estruturada.

[0048] Separação: Ambientes hospedados têm um contexto de segurança atribuído a eles pelo componente base. O contexto de segurança de um ambiente pode conter componentes tais como:

a. contexto DAC, por exemplo, o identificador de código de um ambiente.

b. contexto MAC: no caso de MAC, o contexto MAC de um ambiente consiste em um rótulo de sensibilidade e uma categoria.

[0049] Comunicação: Comunicação entre ambientes é regulada pelo componente base mediante uso de transportes unidirecionais de propriedade do componente base.

a. Transportes proporcionam distribuição de mensagens em seqüência e de forma confiável entre ambientes. Um objeto de sincronização é provido para notificar um ambiente da chegada de dados em um transporte.

b. Controle de acesso: Transportes são objetos de propriedade do componente base, e a capacidade de um ambiente hospedado em ler a partir de, ou gravar em um transporte é regulada pelo modelo de controle de acesso imposto pelo componente base. No caso de DAC, isso é controlado por um ACL no transporte para as ações de "ler" e "gravar". No caso de MAC, isso é controlado pelo rótulo anexado ao transporte. Para prover exportação para dispositivos de múltiplos níveis, o componente base provê implementações de filtros de segurança auxiliares. Filtros de segurança auxiliares (FESF) são funções registradas para o componente base que podem ser anexadas às extremidades de gravar ou ler dos transportes. No caso

de um FESF-de gravar (ler), o FESF de gravar é ativado pelo VMM antes dos dados serem gravados (lidos) no (a partir do) transporte por um VM. Um exemplo de FESF-de gravar é Seal() que é aplicado tipicamente aos dados em um ambiente de alta garantia antes desses dados serem gravados em um ambiente comum (que não é de alta garantia) se MAC estiver em vigor. Se MAC não estiver em vigor, um ambiente terá que codificar dados que ele considera confidenciais antes de enviar os mesmos para um outro ambiente.

[0050] Mensagens: Transportes carregam mensagens através de ambientes diferentes. Uma mensagem é uma gota de dados criada pelo componente base que é estruturada como a seguir:

a. Contexto de segurança atribuído pelo componente base: Contém um contexto DAC e MAC. O contexto DAC contém um identificador-sujeito do ambiente criador. O contexto MAC contém o rótulo de sensibilidade e categoria do ambiente criador.

b. Conteúdo: Dados (incluindo contexto de segurança atribuído por um ambiente criador).

[0051] Abstrações de nível superior: Ambientes (ou aplicações ou outros objetos de software operando em um ambiente) podem implementar uma interface RPC por eles mesmos mediante uso de abstração de comunicação exposta pelo componente base. Isso pode ser um serviço provido por um ambiente, ou simplesmente código de biblioteca. Uma outra abstração que pode ser exposta é uma interface de chamada de soquete entre ambientes, que não precisa ser implementada pelo componente base.

[0052] Gerenciamento de foco: Em muitos dos exemplos de aplicações partilhadas, a chegada de uma mensagem a partir de um ambiente para o outro pode causar uma mudança no proprietário atual do dispositivo seguro de entrada ou saída. Gerenciamento de foco é provido pelo componente base mediante solicitação pelos ambientes -

por exemplo, um ambiente pode solicitar uma mudança de foco mediante chegada de uma mensagem a partir de um outro ambiente. Um ambiente tem, então, uma sessão com o dispositivo seguro de entrada/saída e, então, cede o controle. O componente base pode forçar o término de uma sessão I/O por um ambiente.

Objeto de Dados Exemplar para Uso com Aplicações Partilhadas

[0053] A Figura 7 mostra uma estrutura exemplar de um objeto de dados, que permite que o objeto de dados seja usado com aplicações partilhadas. Como descrito acima, quando uma aplicação é partilhada, uma primeira partição da aplicação pode encontrar um objeto de dados que não pode ser processado por aquela partição, mas que precisa ser enviado para uma segunda partição. No caso quando a primeira partição é a parte "não-segura" da aplicação e a segunda partição é a parte "segura", é preferível que a primeira partição possa reconhecer que o objeto de dados precisa ser enviado para outro lugar para processamento, mesmo se a primeira partição não puder determinar nada mais sobre o objeto (por exemplo, mesmo se a primeira partição não puder ler o conteúdo do objeto). Adicionalmente, em algumas circunstâncias pode ser importante verificar se o objeto de dados foi, na realidade, criado sob circunstâncias seguras e não foi modificado desde então (por exemplo, deve ser possível determinar se os dados foram introduzidos em um ambiente no qual os dados em seu percurso a partir do teclado até o fluxo I/O de uma aplicação são protegidos contra imitação). Dessa forma, também é preferível que o objeto de dados seja representado de uma maneira que permita que sua cadeia de execução seja verificada.

[0054] O objeto 700 de dados inclui um item 702 de dados. O item 702 de dados é o conteúdo substantivo subjacente de que o objeto 700 de dados existe com o propósito de realizar - por exemplo, um documento de processamento de texto secreto (ou uma sua parte),

dados financeiros confidenciais para uma planilha, informação de senha para autenticar um usuário em uma transação financeira, etc.

[0055] O item 702 de dados é envolvido em uma série de invólucros mediante cada camada da arquitetura que gerencia os dados. Cada invólucro tem duas finalidades: (1) o invólucro identifica a identidade (por exemplo, uma aplicação, um ambiente, etc.) que colocou o invólucro no item, que auxilia no encaminhamento posterior do item para aquela identidade; (2) o invólucro sela o item de uma maneira que posteriormente permite que se verifique se o item não foi modificado desde que o mesmo foi envolvido. Por exemplo, se Word-let 608 cria o item 702 de dados, então o Word-let pode anexar uma assinatura digital de item 702 de dados, bem como um cabeçalho que identifica o Word-let como o processador para o qual o item 702 de dados deve ser posteriormente enviado quando ele precisar ser, de alguma forma, manipulado. A assinatura e o cabeçalho desse modo constituem um invólucro 704. Deve ser observado que invólucros não são limitados a assinaturas e cabeçalhos, ou a qualquer modalidade específica, e há diversas técnicas conhecidas que permitem que um item seja identificado, e que permitem que a integridade do item (isto é, não-modificação) seja verificada.

[0056] Como descrito acima, a segurança de um processador tal como Word-let se baseia em uma hierarquia de componentes de confiança - isto é, Word-let é confiável porque se baseia na alta garantia do nexus; o nexus é confiável porque se baseia nas capacidades de isolamento do componente base, etc. Desse modo, cada componente na hierarquia que está no encadeamento que leva à aplicação que cria o item 702 de dados envolve o item em seu próprio invólucro. O item 702 de dados, o qual já está envolvido pelo invólucro do Word-let, é então envolvido pelo invólucro 706 do nexus. Esse item (duplamente envolvido) é então envolvido pelo invólucro 708 do

componente base. Cada invólucro identifica essencialmente o componente de invólucro e constitui também a afirmativa verificável (até o ponto em que o componente de invólucro é confiável) de que o conteúdo envolvido não foi alterado desde que ele foi envolvido.

[0057] Adicionalmente, uma vez que os invólucros são aninhados de acordo com a ordem da hierarquia, cada invólucro é útil para fins de roteamento. Dessa forma, quando o objeto 700 de dados é encontrado por uma aplicação, o invólucro externo 708 identifica o objeto como aquele que precisa ser encaminhado para um outro processador, de modo que o objeto é enviado para o componente base. O componente base usa, então, o invólucro para verificar a integridade do conteúdo dentro do invólucro, e utiliza o invólucro 706 do próximo nível para determinar para qual ambiente o conteúdo deve ser encaminhado (neste caso para o nexus). O nexus verifica então a integridade do conteúdo dentro do invólucro 706, e também utiliza o invólucro 704 para determinar qual objeto de software (neste caso, Word-let) lidará com o conteúdo. O Word-let verifica então a integridade do conteúdo dentro do invólucro 704. Esse conteúdo é o item 702 de dados. Dessa forma, a estrutura de objeto 700 de dados permite que o item 702 de dados seja encaminhado e verificado em cada etapa do encadeamento que leva ao processador que lidará com o item 702 de dados.

Processo Exemplar de Utilização de Aplicação Partilhada

[0058] A Figura 8 mostra um processo exemplar através do qual é utilizada uma aplicação partilhada. Supõe-se, para a finalidade do exemplo da Figura 8, que uma aplicação tem dois processadores, referidos como processador 1 e processador 2.

[0059] Inicialmente, o processador 1 está em operação; durante o tempo em que o processador 1 está em operação, o processador 1 encontra um objeto (802) de dados. Como discutido acima, o

processador 1 pode não ser capaz de ler o objeto de dados, porém pode ser capaz de reconhecer o objeto de dados como algo que precisa ser enviado para algum outro lugar, para processamento. Dessa forma, o processador 1 envia o objeto de dados para o monitor (804) de referência.

[0060] O monitor de referência utiliza o invólucro mais externo do objeto de dados para determinar para onde o objeto de dados precisa ser enviado. Como discutido acima, o monitor de referência (o qual é preferivelmente parte do componente base discutido acima) utiliza também o invólucro mais externo para verificar a integridade daquele conteúdo do invólucro. Neste exemplo, supõe-se que a integridade do conteúdo foi mantida, e que o monitor de referência determina que o objeto de dados precisa ser processado pelo processador 2 (806). (Como discutido acima, o monitor de referência pode não estar diretamente ciente da existência do processador 2, porém mais propriamente pode usar o invólucro para encaminhar o objeto para um ambiente específico, onde o objeto é subseqüentemente encaminhado para o processador 2 pelo ambiente receptor. Uma vez que o uso de invólucros aninhados desta maneira é discutido acima em conexão com a Figura 7, com a finalidade de simplicidade, o exemplo da Figura 8 simplesmente supõe que o objeto de dados pode ser encaminhado ao processador 2 de alguma maneira sem considerar as etapas de encaminhamento intermediárias que podem estar envolvidas no roteamento do objeto para o processador 2).

[0061] Quando tiver sido feita a determinação de que o objeto de dados se destina ao processador 2, é determinado (808) se o processador 2 está em operação. Por exemplo, essa determinação pode ser feita pelo ambiente no qual o processador 2 deve operar (por exemplo, o nexus, nos exemplos anteriores). Se o processador 2 não estiver operando, então o processador 2 é inicializado (810). Após o

processador 2 ter sido inicializado (ou de outra forma for determinado como já estando operando), então o processador 2 é enfocado (isto é, a capacidade de realizar I/O em dispositivos de entrada e saída) se o processador 2 deve lidar com qualquer entrada ou saída.

[0062] Observa-se que os exemplos anteriores foram providos simplesmente com o propósito de explanação e de forma alguma devem ser considerados como limitadores da presente invenção. Embora a presente invenção tenha sido descrita com referência a várias modalidades, entende-se que as palavras que foram usadas aqui são palavras de descrição e ilustração, mais propriamente do que palavras de limitação. Além disso, embora a invenção tenha sido descrita aqui com referência a meios materiais e modalidades específicas, não se pretende que a invenção seja limitada aos aspectos específicos aqui revelados; mais propriamente, a invenção se estende a todas as estruturas, métodos e usos funcionalmente equivalentes, tais como compreendidos no escopo das reivindicações anexas. Aqueles versados na técnica, tendo o benefício dos ensinamentos dessa especificação, podem realizar várias modificações na mesma, e podem ser feitas alterações sem se afastar do escopo e espírito da invenção, em seus aspectos.

REIVINDICAÇÕES

1. Sistema que gerencia o particionamento de uma aplicação, caracterizado por compreender:

uma camada base (508) que hospeda a operação de um primeiro ambiente (502(1))e de um segundo ambiente (502(2)), a aplicação compreendendo:

um primeiro objeto de software (504(1), 504(2)) que executa no primeiro ambiente (502(1)), o primeiro objeto de software (504(1), 504(2)) manipulando uma pluralidade de dados (506) e incluindo lógica para identificar um primeiro dado da pluralidade de dados como não processável pelo objeto de software; e

um segundo objeto de software (504(3), 504(4)) que executa no segundo ambiente (502(2)) e que processa o primeiro dado da pluralidade de dados de uma forma que resiste à violação do primeiro dado da pluralidade de dados,

a camada base compreendendo ou hospedando uma lógica (501) que recebe o primeiro dado da pluralidade de dados a partir do objeto de software e encaminha o primeiro dado da pluralidade de dados para o segundo ambiente (502(2)).

2. Sistema, de acordo com a reivindicação 1, caracterizado pelo fato de que o primeiro objeto de software (504(1), 504(2)) faz com que uma representação do primeiro dado da pluralidade de dados seja exibida em um dispositivo de vídeo, a representação compreendendo um ou mais sinais indecifráveis.

3. Sistema, de acordo com a reivindicação 2, caracterizado pelo fato de que um ou mais sinais indecifráveis ou são: (i) do mesmo tamanho uns dos outros, ou (ii) de tamanhos que não são relacionados ao conteúdo do primeiro dado da pluralidade de dados.

4. Sistema, de acordo com a reivindicação 1, caracterizado pelo fato de que a resistência à violação provida pelo segundo objeto

de software (504(3), 504(4)) compreende o segundo ambiente (502(2)) resistindo à interferência com a exibição do primeiro dado da pluralidade de dados mediante gravação de uma representação do primeiro dado da pluralidade de dados em uma memória de vídeo associada a um dispositivo de vídeo de modo a fazer com que a representação substitua qualquer imagem em um local no dispositivo de exibição no qual a representação deve ser exibida.

5. Sistema, de acordo com a reivindicação 1, caracterizado pelo fato de que o primeiro dado da pluralidade de dados é introduzido em um teclado, e em que a resistência à violação provida pelo segundo objeto de software (504(3), 504(4)) compreende resistir à violação do primeiro dado da pluralidade de dados em trânsito a partir do teclado para um fluxo de entrada do segundo objeto de software (504(3), 504(4)).

6. Sistema, de acordo com a reivindicação 5, caracterizado pelo fato de que a segunda aplicação marca o primeiro dado da pluralidade de dados para prevenir violação subseqüente do primeiro dado da pluralidade de dados.

7. Sistema, de acordo com a reivindicação 6, caracterizado pelo fato de que o segundo ambiente (502(2)) marca o primeiro dado da pluralidade de dados e a assinatura criada pela segunda aplicação como uma indicação de que o primeiro dado da pluralidade de dados e a assinatura foram criados no segundo ambiente (502(2)).

8. Sistema, de acordo com a reivindicação 1, caracterizado pelo fato de que a camada base compreende um componente que atribui um primeiro identificador ao segundo ambiente (502(2)).

9. Sistema, de acordo com a reivindicação 8, caracterizado pelo fato de que o primeiro dado da pluralidade de dados inclui, ou é acompanhado pelo, primeiro identificador e um segundo identificador que identifica o segundo objeto de software (504(3), 504(4)).

10. Sistema, de acordo com a reivindicação 1, caracterizado pelo fato de que o primeiro ambiente (502(1)) é associado a uma primeira especificação que descreve o comportamento do primeiro ambiente (502(1)), em que o segundo ambiente (502(2)) é associado a uma segunda especificação que descreve o comportamento do segundo ambiente (502(2)), em que há um nível maior de garantia de que o segundo ambiente (502(2)) estará em conformidade com a segunda especificação do que, que o primeiro ambiente (502(1)) estará em conformidade com a primeira especificação.

11. Sistema, de acordo com a reivindicação 10, caracterizado pelo fato de que o segundo objeto de software (504(3), 504(4)) se baseia no comportamento do segundo ambiente (502(2)) para resistir à violação do primeiro dado da pluralidade de dados.

12. Sistema, de acordo com a reivindicação 1, caracterizado pelo fato de que a camada base é o segundo ambiente (502(2)), ou está incluída no segundo ambiente (502(2)).

13. Método de um primeiro objeto de software (504(1), 504(2)), que executa em um primeiro ambiente (502(1)), manipulando dados aos quais se aplica uma política, o método sendo caracterizado por compreender:

o primeiro objeto de software (504(1), 504(2)) encontrando os dados;

o primeiro objeto de software (504(1), 504(2)) determinando que os dados não são processáveis pelo primeiro objeto de software (504(1), 504(2));

o primeiro objeto de software (504(1), 504(2)) fazendo com que os dados sejam providos a um segundo objeto de software (504(3), 504(4)) que executa em um segundo ambiente (502(2)) que proporciona um primeiro nível de garantia de que as ações realizadas

no segundo ambiente (502(2)) serão realizadas corretamente, em que o segundo objeto de software (504(3), 504(4)) processa os dados de uma maneira que utiliza a garantia para resistir à violação dos dados mediante ações que surgem fora do segundo ambiente (502(2)).

14. Método, de acordo com a reivindicação 13, caracterizado pelo fato de que a resistência à violação compreende uma resistência a uma mudança nos dados.

15. Método, de acordo com a reivindicação 14, caracterizado pelo fato de que os dados devem ser exibidos em um dispositivo de exibição visual, e em que a resistência à violação compreende exibir uma representação dos dados em um local no dispositivo de exibição visual e substituir qualquer imagem que não seja a representação que é renderizada no local.

16. Método, de acordo com a reivindicação 13, caracterizado pelo fato de que o primeiro objeto de software (504(1), 504(2)) faz com que uma representação dos dados seja exibida em um dispositivo de exibição visual, a representação compreendendo um ou mais sinais indecifráveis.

17. Método, de acordo com a reivindicação 16, caracterizado pelo fato de que as representações ou são: (i) do mesmo tamanho umas das outras, ou (ii) de tamanhos que não são relacionados ao conteúdo do primeiro dado da pluralidade de dados.

18. Método, de acordo com a reivindicação 16, caracterizado pelo fato de que o primeiro objeto de software (504(1), 504(2)) ou o segundo objeto de software (504(3), 504(4)), ou uma combinação do primeiro objeto de software (504(1), 504(2)) e o segundo objeto de software (504(3), 504(4)), faz com que itens exibidos no dispositivo de exibição visual sejam mudados em pelo menos um aspecto para permitir que se visualize uma imagem dos dados produzidos pelo segundo objeto de software (504(3), 504(4)).

19. Método, de acordo com a reivindicação 14, caracterizado pelo fato de que os dados são providos utilizando-se um teclado, e em que a resistência à violação compreende resistir a uma mudança nos dados em trânsito a partir do teclado para o fluxo de entrada do segundo objeto de software (504(3), 504(4)).

20. Método, de acordo com a reivindicação 13, caracterizado pelo fato de que a política de segurança especifica que os dados devem ser manipulados pelo segundo objeto de software (504(3), 504(4)).

21. Método, de acordo com a reivindicação 13, caracterizado pelo fato de que os dados incluem, ou são associados a, um primeiro rótulo que identifica o segundo ambiente (502(2)) como um local no qual os dados devem ser processados.

22. Método, de acordo com a reivindicação 21, caracterizado pelo fato de que os dados incluem, ou são associados a um segundo rótulo o qual identifica o segundo objeto de software (504(3), 504(4)) como um processador para os dados, e em que o segundo ambiente (502(2)) encaminha os dados para o segundo objeto de software (504(3), 504(4)) com base no segundo rótulo.

23. Método, de acordo com a reivindicação 13, caracterizado pelo fato de que o segundo ambiente (502(2)) é associado a uma primeira especificação que descreve o comportamento do segundo ambiente (502(2)), e em que a garantia provê que o segundo ambiente (502(2)) estará em conformidade com a especificação.

24. Método, de acordo com a reivindicação 13, caracterizado pelo fato de que o primeiro ambiente (502(1)) é associado a uma segunda especificação que descreve o comportamento do primeiro ambiente (502(1)), e em que o primeiro ambiente (502(1)) provê um segundo nível de garantia de que as

ações realizadas no primeiro ambiente (502(1)) serão realizadas corretamente, o segundo nível de garantia sendo relativamente inferior ao primeiro nível de garantia.

25. Sistema que suporta o particionamento de uma aplicação em pelo menos um primeiro objeto de software (504(1), 504(2)) e um segundo objeto de software (504(3), 504(4)), o sistema hospedando um primeiro ambiente (502(1)) e um segundo ambiente (502(2)), o primeiro objeto de software (504(1), 504(2)) executando no primeiro ambiente (502(1)), o segundo objeto de software (504(3), 504(4)) executando no segundo ambiente (502(2)), o sistema sendo caracterizado por compreender uma interface de programação de aplicação que expõe pelo menos um dos seguintes métodos:

um primeiro método que recebe a partir do primeiro objeto de software (504(1), 504(2)) um primeiro objeto de dados que compreende: (i) dados processáveis pelo segundo objeto de software (504(3), 504(4)), e (ii) um primeiro identificador atribuído pelo sistema ao segundo ambiente (502(2)); e que encaminha o primeiro objeto de dados para o segundo ambiente (502(2)) com base no primeiro identificador;

um segundo método que cria um segundo objeto de dados que compreende: (i) dados processáveis pelo segundo objeto de software (504(3), 504(4)); (ii) o primeiro identificador; (iii) dados de autenticação que permitem uma determinação subseqüente de que o segundo objeto de dados não foi violado desde que foi criado pelo segundo método;

um terceiro método que recebe, a partir do primeiro ambiente (502(1)), um segundo identificador associado ao segundo objeto de software (504(3), 504(4)), e que indica que uma instância do segundo objeto de software (504(3), 504(4)) seja criada; e

um quarto método que recebe, a partir do primeiro

ambiente (502(1)) de software: (i) um terceiro objeto de dados, e (ii) um terceiro identificador associado ao primeiro objeto de software (504(1), 504(2)), e que indica que uma instância do primeiro objeto de software (504(1), 504(2)) seja criada com base no fato de ter recebido o terceiro identificador, e que indica que o primeiro objeto de software (504(1), 504(2)) opere sobre o terceiro objeto de dados.

26. Sistema, de acordo com a reivindicação 25, caracterizado pelo fato de que o primeiro ambiente (502(1)) é associado a uma primeira especificação que descreve o comportamento do primeiro ambiente (502(1)), em que o segundo ambiente (502(2)) é associado a uma segunda especificação que descreve o comportamento do segundo ambiente (502(2)), em que há um primeiro nível de garantia de que o primeiro ambiente (502(1)) estará em conformidade com a primeira especificação, em que há um segundo nível de garantia de que o segundo ambiente (502(2)) estará em conformidade com a segunda especificação, e em que o segundo nível de garantia é relativamente maior do que o primeiro nível de garantia.

27. Sistema, de acordo com a reivindicação 26, caracterizado pelo fato de que o segundo software provê garantia de que o segundo objeto de software (504(3), 504(4)) protegerá os dados, a garantia sendo provida pelo menos em parte pelo fato de se basear no comportamento do segundo ambiente (502(2)).

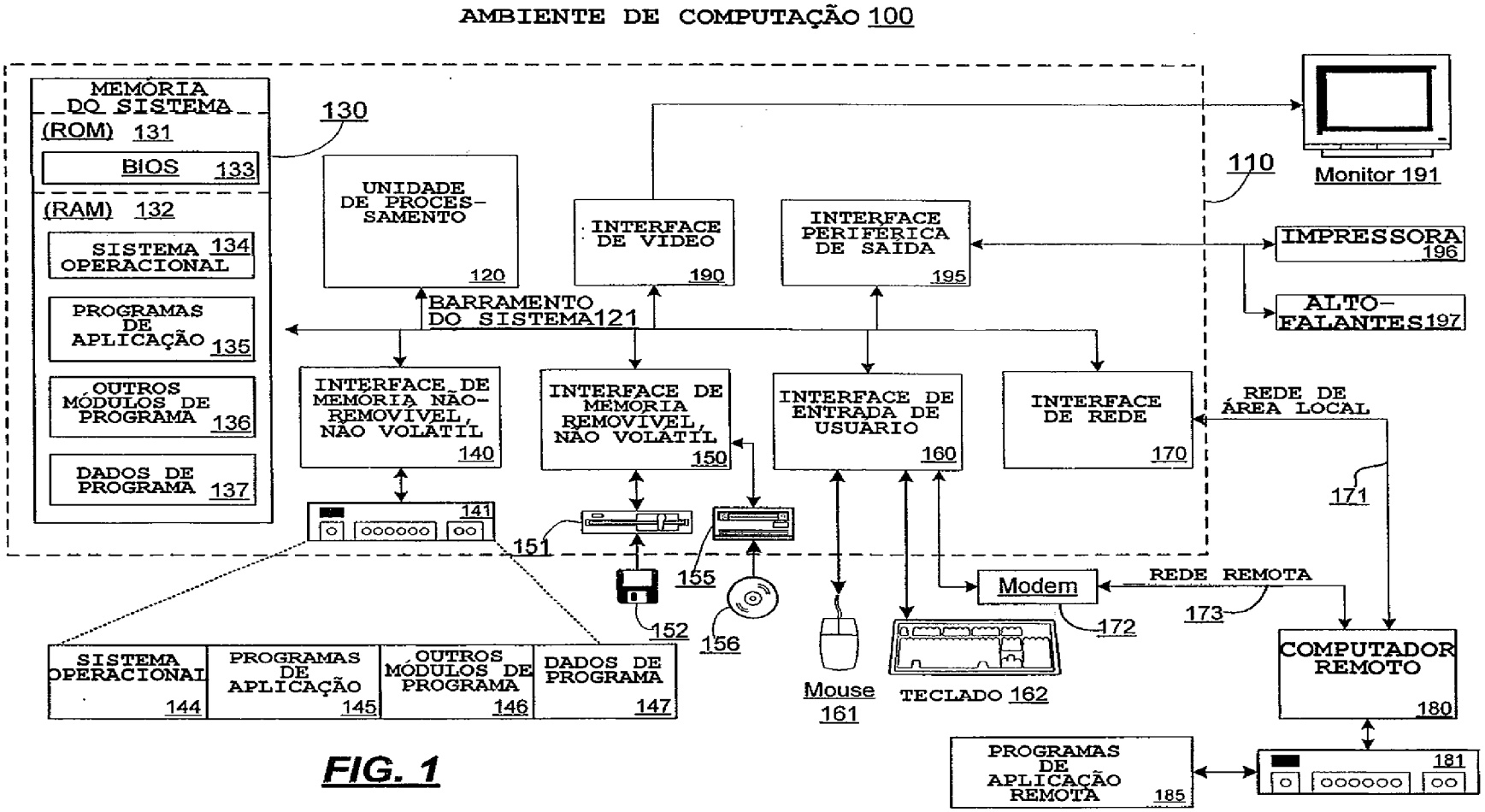
AMBIENTE DE COMPUTAÇÃO 100
MEMÓRIA DO SISTEMA
(ROM) 131
BIOS 133
(RAM) 132
SISTEMA OPERACIONAL 134
PROGRAMAS DE APLICAÇÃO 135
OUTROS MÓDULOS DE PROGRAMA 136
DADOS DE PROGRAMA 137
130
UNIDADE DE PROCESSAMENTO 120
INTERFACE DE VIDEO 190
INTERFACE PERIFÉRICA DE SAÍDA 195
110
Monitor 191
IMPRESSORA 196
ALTO-FALANTES 197
BARRAMENTO DO SISTEMA 121
INTERFACE DE MEMÓRIA NÃO-REMOVÍVEL, NÃO VOLÁTIL 140
INTERFACE DE MEMÓRIA REMOVÍVEL, NÃO VOLÁTIL 150
INTERFACE DE ENTRADA DE USUÁRIO 160
INTERFACE DE REDE 170
REDE DE ÁREA LOCAL
171
141
151
155
152
156
Modem
172
REDE REMOTA
173
SISTEMA OPERACIONAL 144
PROGRAMAS DE APLICAÇÃO 145
OUTROS MÓDULOS DE PROGRAMA 146
DADOS DE PROGRAMA 147
Mouse 161
TECLADO 162
COMPUTADOR REMOTO 180
PROGRAMAS DE APLICAÇÃO REMOTA 185
181

FIG. 1

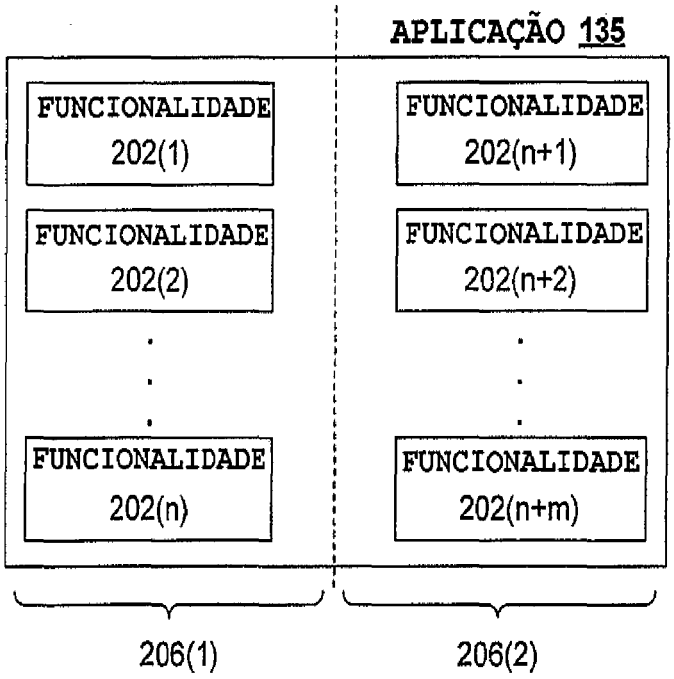
APLICAÇÃO 135
FUNCIONALIDADE 202(1)
FUNCIONALIDADE 202(2)
FUNCIONALIDADE 202(n)
FUNCIONALIDADE 202(n+1)
FUNCIONALIDADE 202(n+2)
FUNCIONALIDADE 202(n+m)
206(1)
206(2)

FIG. 2

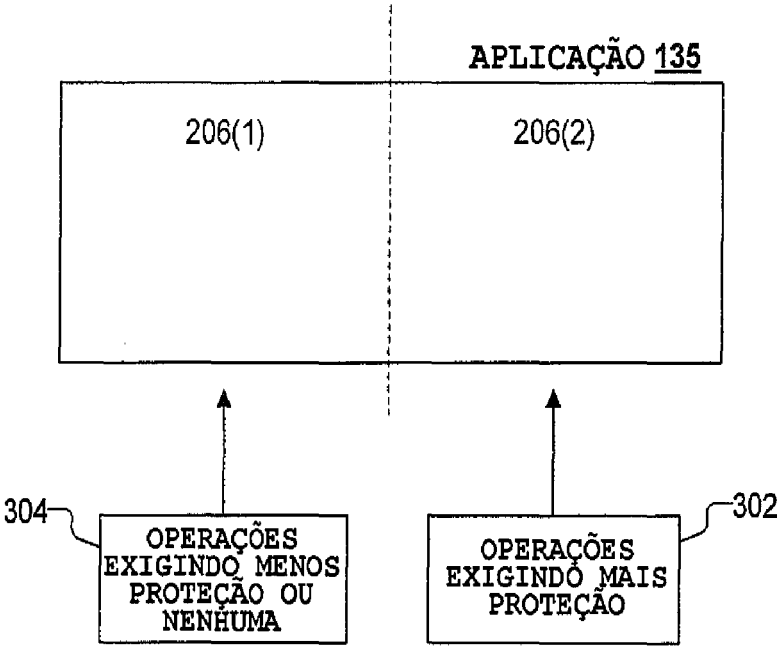
APLICAÇÃO 135
206(1)
206(2)
304
OPERAÇÕES EXIGINDO MENOS PROTEÇÃO OU NENHUMA
OPERAÇÕES EXIGINDO MAIS PROTEÇÃO
302

FIG. 3

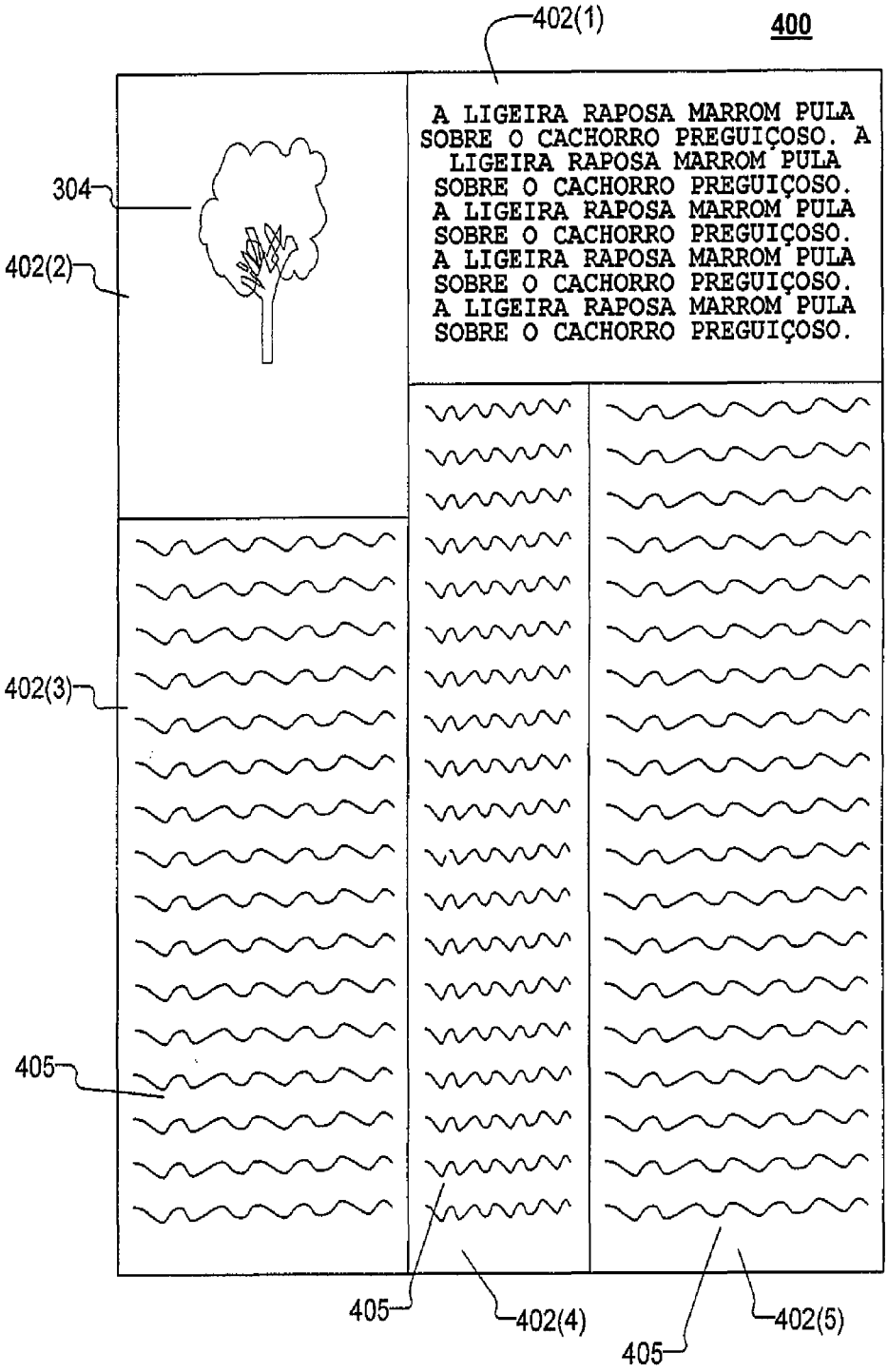
402(1)
400
A LIGEIRA RAPOSA MARROM PULA SOBRE O CACHORRO PREGUIÇOSO. A LIGEIRA RAPOSA MARROM PULA SOBRE O CACHORRO PREGUIÇOSO. A LIGEIRA RAPOSA MARROM PULA SOBRE O CACHORRO PREGUIÇOSO. A LIGEIRA RAPOSA MARROM PULA SOBRE O CACHORRO PREGUIÇOSO. A LIGEIRA RAPOSA MARROM PULA SOBRE O CACHORRO PREGUIÇOSO.
304
402(2)
402(3)
405
405
402(4)
405
402(5)

FIG. 4

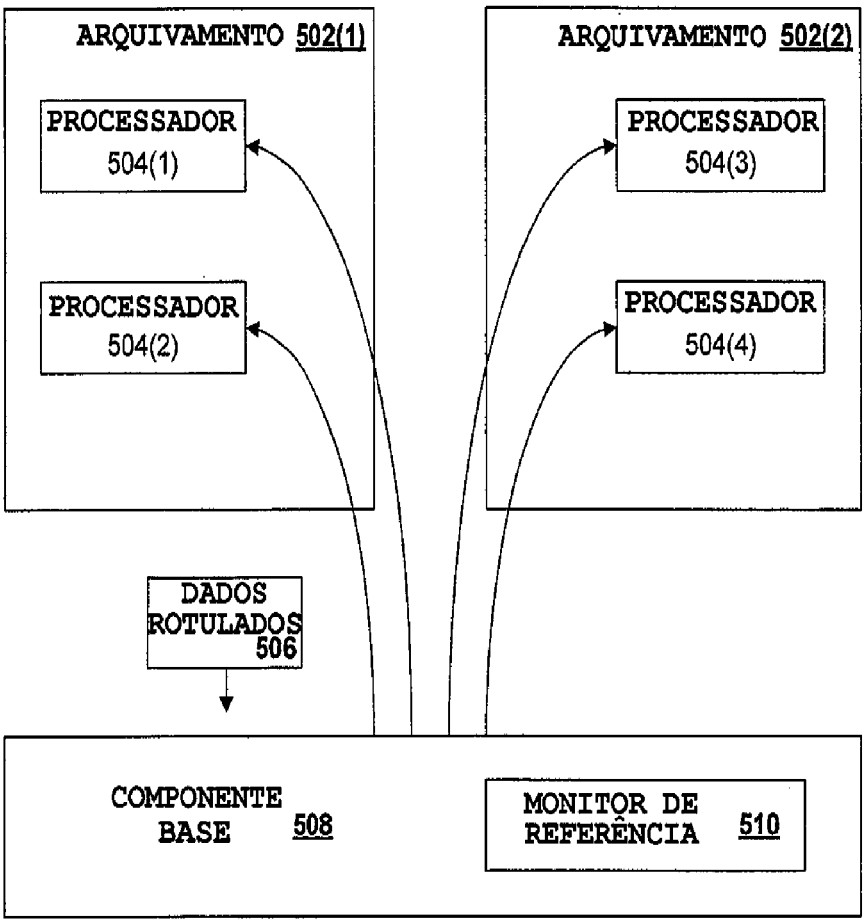
ARQUIVAMENTO 502(1)
PROCESSADOR 504(1)
PROCESSADOR 504(2)
ARQUIVAMENTO 502(2)
PROCESSADOR 504(3)
PROCESSADOR 504(4)
DADOS ROTULADOS 506
COMPONENTE BASE 508
MONITOR DE REFERÊNCIA 510

FIG. 5

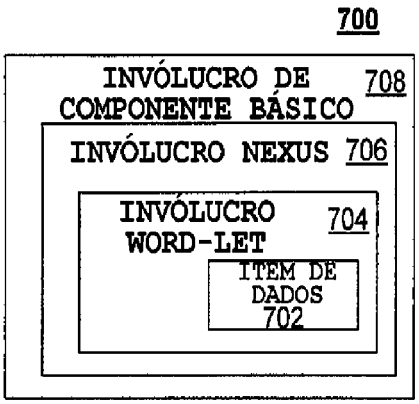
700
INVÓLUCRO DE COMPONENTE BÁSICO 708
INVÓLUCRO NEXUS 706
INVÓLUCRO WORD-LET 704
ITEM DE DADOS 702

FIG. 7

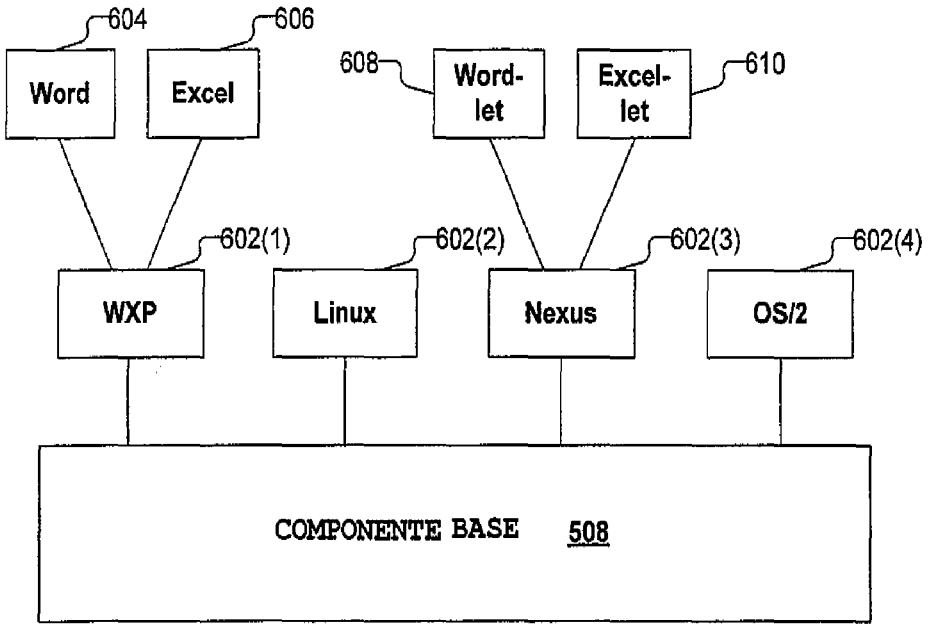
604
606
608
610
Word
Excel
Word-let
Excel-let
602(1)
602(2)
602(3)
602(4)
WXP
Linux
Nexus
OS/2
COMPONENTE BASE 508

FIG. 6