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

(21) **PI0710036-1 A2**

BRPI0710036A2

(22) Data de Depósito: 21/02/2007
(43) Data da Publicação: 02/08/2011
(RPI 2117)

(51) Int.Cl.:
G06Q 99/00 2006.01
G06F 9/44 2006.01

(54) Título: **MODELO DECLARATIVO PARA CONTROLE E SIMULTÂNEO ATRAVÉS DE LINHAS DE EXECUÇÃO LEVES**

(30) Prioridade Unionista: 30/03/2006 US 11/393,966

(73) Titular(es): Microsoft Corporation

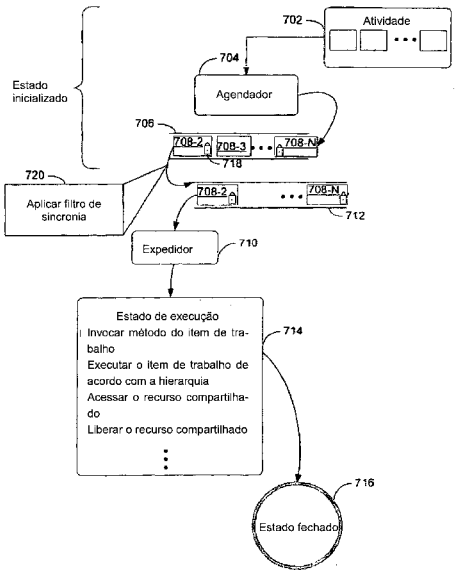
(72) Inventor(es): Akash J. Sagar, Bob Schmidt, Dharma Shukla

(74) Procurador(es): Nellie Anne Daniel Shores

(86) Pedido Internacional: PCT US2007004633 de 21/02/2007

(87) Publicação Internacional: WO 2007/120390de 25/10/2007

(57) Resumo: MODELO DECLARATIVO PARA CONTROLE SIMULTÂNEO ATRAVÉS DE LINHAS DE EXECUÇÃO LEVES E divulgada a implementação da sincronia entre linhas de execução em um fluxo de trabalho. Uma área de memória armazena uma pluralidade de itens de trabalho em uma fila do agendador. Os itens de trabalho são associados com uma atividade no fluxo de trabalho,e cada item de trabalho é associado com uma linha de execução. Um processador é configurado para atribuir uma rotina de tratamento de sincronia a cada um dos itens de trabalho.A rotina de tratamento de sincronia indica um recurso compartilhado em particular a ser acessado pelos itens de trabalho. Um valor com sinal é computado para cada item de trabalho com base na rotina de tratamento de sincronia atribuida e nos itens de trabalho em uma hierarquia na atividade. Os itens de trabalho são ordenados em uma fila de sincronia combase no valor com sinal associado com cada item de trabalho. O processador executa sequencialmente cada um dos itens de trabalho armazenados na fila de sincronia para tornarserial o acesso ao recurso compartilhado em particular e efetuar uma execução síncrona das linhas de execução associadas com os itens de trabalho.



"MODELO DECLARATIVO PARA CONTROLE SIMULTÂNEO ATRAVÉS DE LINHAS DE EXECUÇÃO LEVES"

Antecedentes da invenção

Programas orientados a processo ou centralizados em processo evoluíram para habilitar o processamento de instruções complexas que modelam interações do mundo real entre agentes autônomos. Sistemas existentes tentam mapear problemas empresariais em fluxos de trabalho de alto nível pela modelagem do problema empresarial. Entretanto, fluxos de trabalho do mundo real variam em uma multiplicidade de dimensões, tais como, (a) complexidade de execução e de modelagem, (b) conhecimento da estrutura do fluxo no momento do projeto, (c) estaticamente definido ou ad-hoc / dinâmico, (d) facilidade da autoria e da edição do fluxo em vários pontos no seu ciclo de vida e (e) fraca ou forte associação da lógica empresarial com o processo central do fluxo de trabalho. Modelos existentes deixam de conciliar estes fatores.

Adicionalmente, a maior parte dos modelos de fluxo de trabalho existentes é baseada tanto em abordagens baseadas em linguagem (por exemplo, BPEL4WS, XLANG/S e WSFL) quanto em abordagens baseadas em aplicação. Abordagens baseadas em linguagem são linguagens de fluxo de trabalho de alto nível com um conjunto fechado de construções pré-definidas que ajudam a modelar o processo do fluxo de trabalho para o usuário / programador. As linguagens de fluxo de trabalho portam toda a informação semântica para o conjunto fechado de construções para habilitar o usuário a construir um modelo de fluxo de trabalho. Entretanto, as linguagens não são extensíveis pelos desenvolvedores e representam um conjunto fechado de elementos básicos que constitui o modelo do fluxo de trabalho. As linguagens são amarradas ao compilador da linguagem embarcado pelo revendedor do sistema de fluxo de trabalho. Somente o revendedor do produto do sistema de fluxo de trabalho pode estender o modelo pela extensão da linguagem com um conjunto inédito de construções em uma versão futura do produto. Freqüentemente, isto exige atualização do compilador associado com a linguagem. Além do mais, usualmente, as linguagens não expõem ou definem declarativamente funções ou operações que podem ser usadas de forma fácil e eficiente por outros programas.

Abordagens com base em aplicação são aplicações que têm capacidades de fluxo de trabalho na aplicação para resolver um problema específico de domínio. Estas aplicações não são verdadeiramente extensíveis nem têm um modelo programável.

Além do mais, com as abordagens existentes, as questões de complexidade, de conhecimento prévio, de fluxos de trabalho dinâmicos, de facilidade de autoria e de força de associações com a lógica empresarial e com fluxos de trabalho centrais não são adequadamente abordadas. Não há estruturas projetistas de fluxo de trabalho extensíveis, customizáveis e ré-hospedáveis disponíveis para construir projetistas de fluxo de trabalho visual

para modelar diferentes classes de fluxos de trabalho. Sistemas existentes carecem de uma experiência de projeto de fluxo de trabalho no estilo rápido desenvolvimento de aplicação (RAD), que permite que usuários projetem graficamente o processo do fluxo de trabalho e associem a lógica empresarial em uma linguagem de programação da escolha do desenvolvedor.

Também, os processos de fluxo de trabalho lidam com preocupações de atalho ortogonal e confuso que abrangem múltiplas etapas de um modelo de processo de fluxo de trabalho. Por exemplo, embora partes dos processos de fluxo de trabalho sejam projetadas para participar de transações de longo prazo, outras partes do mesmo processo são projetadas para execução simultânea ou para acessar um recurso compartilhado. Em função das deficiências de projeto, sistemas existentes deixam de fornecer intercalação das linhas de execução que habilitam usuários a projetar execução síncrona ou intercalada das atividades. Ainda outras partes do mesmo processo de fluxo de trabalho exigem rastreamento, embora outras partes tratem de exceções em nível empresarial ou da aplicação. Há uma necessidade de aplicar certos comportamentos a uma ou mais partes de um processo do fluxo de trabalho.

Algumas abordagens da modelagem do fluxo de trabalho são impraticáveis, já que elas exigem uma completa descrição com base no fluxo de todo o processo empresarial, incluindo todas as exceções e intervenções humanas. Algumas destas abordagens fornecem funcionalidade adicional à medida que as exceções surgem, embora outras abordagens empreguem exclusivamente uma abordagem baseada em restrição em vez de uma abordagem baseada no fluxo para modelar um processo empresarial. Sistemas existentes implementam tanto a abordagem baseada no fluxo quanto a abordagem baseada em restrição. Tais sistemas são muito inflexíveis para modelar muitas situações empresariais comuns. Estes sistemas também carecem da capacidade de tratar de forma assíncrona exceções ou cancelamentos.

Sumário da invenção

Modalidades da invenção executam itens de trabalho de forma síncrona acessando um recurso compartilhado que usa um identificador de sincronia para uma linha de execução. Com um controle simultâneo eficiente através das linhas de execução leves, aspectos da invenção usam o identificador de sincronia para identificar o recurso compartilhado de maneira tal que a linha de execução para o item de trabalho seja habilitada a acessar o recurso compartilhado sem interferir em acessos por outras linhas de execução associadas com outros itens de trabalho.

Este Sumário é fornecido para introduzir uma seleção de conceitos de uma forma simplificada que é adicionalmente descrita na seguinte Descrição Detalhada. Não pretende-se que este Sumário identifique recursos chaves ou recursos essenciais do assunto em

questão reivindicado, nem pretende-se que seja usado como um auxílio na determinação do escopo do assunto em questão reivindicado.

Outros recursos ficarão, em parte, aparentes e serão, em parte, salientados a seguir.

Descrição resumida dos desenhos

A figura 1 é um diagrama de blocos que ilustra um paradigma de programação existente.

A figura 2 é um diagrama de blocos exemplar que ilustra uma virtualização de uma estrutura de projeto de fluxo de trabalho de acordo com uma modalidade da invenção.

A figura 3 é um diagrama exemplar que ilustra um fluxo de trabalho exemplar de acordo com uma modalidade da invenção.

A figura 4 é um diagrama que ilustra um sistema para processar atividades do fluxo de trabalho de acordo com uma modalidade da invenção.

A figura 5 é um diagrama que ilustra uma estrutura hierárquica de uma atividade do fluxo de trabalho de acordo com uma modalidade da invenção.

A figura 6 é um diagrama que ilustra um autômato de estado exemplar que descreve estados de processamento dos itens de trabalho associados com uma atividade de acordo com uma modalidade da invenção.

A figura 7A é um diagrama de blocos que ilustra uma execução síncrona dos itens de trabalho em uma atividade em um fluxo de trabalho de acordo com uma modalidade da invenção.

A figura 7B é um diagrama que ilustra a classificação de uma fila de sincronia de acordo com uma modalidade da invenção.

A figura 8 é um fluxograma que ilustra um método para executar de forma síncrona atividades que acessam um recurso compartilhado em particular de acordo com uma modalidade da invenção.

A figura 9 é um diagrama de blocos que ilustra uma mídia legível por computador exemplar na qual aspectos da invenção podem ser armazenados.

O Apêndice A ilustra um conjunto de operações SynchronizationScope que mostra uma implementação exemplar da sincronia ou intercalação do estado compartilhado através das linhas de execução de uma maneira declarativa.

O Apêndice B ilustra um conjunto exemplar de etapas de sincronia da execução da linha de execução de acordo com a hierarquia da atividade.

Caracteres de referência correspondentes indicam partes correspondentes por todos os desenhos.

Descrição detalhada

Primeiro, em relação à figura 1, um diagrama de blocos ilustra um paradigma de

programação existente para projetar programas para atividades centralizadas em processo, tal como um fluxo de trabalho. Por exemplo, o diagrama mostra um modelo de virtualização de três níveis do paradigma de programa existente com um nível de um ambiente de execução gerenciado que é o nível mais alto e uma unidade de processamento que é o nível mais baixo. Neste sistema de projeto de programação, mesmo no nível do ambiente de execução gerenciada, programas, especialmente programas centralizados em processo que tratam processos do fluxo de trabalho, carecem de capacidade e de eficiência para conciliar interações complexas entre processos em um fluxo de trabalho.

Versados na técnica sabem que certas restrições estão associadas com software ou programas de aplicação de projeto. Neste exemplo, na gravação de um programa de software do sistema operacional 104, os códigos ou rotinas de programação dependem do tipo ou da configuração das unidades de processamento 102, sendo específico, ao tipo de arquitetura computacional (por exemplo, compatível com IBM®, computadores APPLE® ou com outros sistemas), ou de outras restrições. Além do mais, tipicamente, linguagens de programação precisam identificar e utilizar precisamente estrutura de dados, tais como pilhas, árvore binária, base de linha de execução ou outras estruturas específicas de hardware, para o sistema operacional 104 funcionar apropriadamente.

Ao lidar com processos de fluxo de trabalho complexos, aplicações existentes usam um conceito de um ambiente de execução gerenciado 106 (por exemplo, um ambiente de tempo de execução em que programas podem compartilhar funções ou classes orientadas a objetos comuns) nos quais programas escritos em uma linguagem de programação podem chamar funções em outros programas escritos em uma linguagem de programação diferente. Em um ambiente de execução como este, estes programas em diferentes linguagens de programação são compilados em uma linguagem intermediária de maneira tal que o ambiente de execução gerenciado 106 possa expor parâmetros, argumentos ou esquemas ou funções a diferentes programas, para que os programas possam interagir uns com os outros.

Embora este ambiente de execução 106 crie um ambiente de comunicação comum entre programas, o ambiente de execução 106 inclui várias exigências rígidas que podem não ser adequadas para tratar a complexidade e a capacidade dos programas centralizados em processo. Por exemplo, o ambiente de execução 106 exige que programas sejam confirmados em um formato de arquivo específico. O ambiente de execução 106 também exige que funções ou operações nos programas usem um conjunto fixo de funções ou uma classe de funções definidos pelo ambiente de execução 106.

Modalidades da invenção construídas em uma fundação ou estrutura extensível 202 da figura 2 superam as deficiências do modelo de programação existente. Pela permissão de que programas escritos em qualquer linguagem de programação sejam compostos

em qualquer formato de arquivo, aspectos da invenção habilitam desenvolvedores de programa a projetar programas com funções específicas sem comprometer suas funcionalidades e especificações. Pela definição das atividades, tais como tarefas ou processos do fluxo de trabalho, como a classe base a ser executada na estrutura do fluxo de trabalho, desenvolvedores podem construir de forma fácil e eficiente códigos de operação específicos de domínio (por exemplo, ambientes de execução específicos, tais como programas na indústria da saúde, na indústria financeira ou congêneres) (doravante "op-code") sem se unir ao rígido, embutido em código, inflexível e fixo conjunto de funções ou de classes de atividades no ambiente de execução existente. Além do mais, a fundação do fluxo de trabalho que incorpora aspectos da invenção é um tempo de execução com base em continuação disposto em camadas no topo de todas as estruturas existentes (por exemplo, tanto um ambiente de execução gerenciado, um ambiente de sistema operacional quanto um nível de unidade de processamento de hardware).

Aspectos da invenção liberam a restrição de definir atividades em um formato de arquivo em particular pela habilitação de projetos de fluxo de trabalho em qualquer maneira ou representação (por exemplo, um fluxograma, um diagrama, uma descrição numerada ou congêneres), contanto que as atividades no fluxo de trabalho possam ser construídas a partir da representação dos projetos do fluxo de trabalho.

A figura 3 ilustra uma vista simplista de um fluxo de trabalho 300 de acordo com uma modalidade da invenção. Por exemplo, o fluxo de trabalho 300 pode ser um fluxo de trabalho para processar um pedido de compra, e este fluxo de trabalho 300 de pedido de compra pode incluir processos ou atividades tais como receber um pedido de compra, transmitir confirmação a um cliente, aprovar o pedido de compra por um gerente ou congêneres. Adicionalmente, estas atividades podem ser seqüenciadas de maneira tal que algumas possam ser realizadas ao mesmo tempo em que outras, embora algumas outras possam ser realizadas somente mediante a conclusão das atividades.

O fluxo de trabalho 300 pode começar a partir de um ponto de início 302. Por exemplo, o ponto de início 302 para o fluxo de trabalho do pedido de compra pode ser receber uma ordem de um cliente. O fluxo de trabalho 300 também pode incluir uma declaração condicional 304 (tais como uma "declaração SE" ou uma "declaração ENQUANTO"), e ele pode ser subdividido em declarações condicionais adicionais 306 e 308. O fluxo de trabalho 300 também pode incluir uma estrutura paralela 310, que inclui adicionalmente uma ou mais atividades 312. Por exemplo, a estrutura paralela 310 pode indicar que atividades, tais como verificar o inventário e atualizar verificação de transportador disponível, podem ser processadas em paralelo. No exemplo mostrado, atividades, tais como "Transmitir Correio Eletrônico" e "Receber Aprovação", podem ser processadas em paralelo. Uma caixa "deixar atividades aqui" 316 indica que um usuário pode adicionar ou complementar adicionalmente mais

atividades no fluxo de trabalho 300. Para completar o fluxo de trabalho 300, os processos ou atividades concluirão em uma etapa ou ponto de conclusão 314.

Em uma modalidade, as atividades podem ser hierarquicamente arranjadas em uma estrutura de árvore (veja figura 5) 500. Por exemplo, um método da atividade está em um nó raiz 502 com dois nós filhos ou folhas 504 e 506. Os métodos da atividade nos nós filhos 504 e 506 (por exemplo, work item_1 e work item_2, respectivamente) podem ser executados de acordo com a estrutura hierárquica. Além do mais, os nós filhos 504 e 506 também podem incluir outros nós filhos com respectivos itens de trabalho a ser executados.

Em uma outra modalidade, atividades incluem um ou mais dos tipos seguintes: uma atividade simples, atividade de recipiente e atividade raiz. Nesta modalidade, há uma atividade raiz no modelo, e nenhuma ou qualquer quantidade de atividades simples ou de atividades de recipiente no interior da atividade raiz. Uma atividade de recipiente pode incluir atividades simples ou de recipiente. Todo o processo de fluxo de trabalho pode ser usado como uma atividade para construir processos de fluxo de trabalho de ordem superior. Adicionalmente, uma atividade pode ser interrompível ou não interrompível. Uma atividade composta não interrompível não inclui atividades interrompíveis. Uma atividade não interrompível carece de serviços que podem fazer com que a atividade seja bloqueada. Além do mais, atividades podem ser atividades primitivas ou agrupadas em uma atividade composta. Uma atividade primitiva ou básica não tem subestrutura (por exemplo, atividades de filho) e, assim, é um nó folha em uma estrutura de árvore. Uma atividade composta contém subestrutura (por exemplo, é o pai de uma ou mais atividades de filho).

Além do mais, na execução de atividades e dos itens de trabalho incluídos nas atividades, a estrutura de fluxo de trabalho define um contexto ou ambiente de execução que é o escopo ou o limite para cada um dos itens de trabalho. Este escopo ou limite inclui e expõe informação (por exemplo, na forma de dados, metadados ou congêneres), tais como os dados ou recursos compartilhados a ser acessados pelos itens de trabalho, propriedades associadas, identificadores, restrições e interações entre agentes autônomos. Estes escopos podem ser estruturados hierarquicamente. Também, cada atividade pode ser configurada por um código de usuário em qualquer linguagem de programação que suporta a estrutura fundamental gerenciada. Por exemplo, o código de usuário pode representar lógica empresarial ou de aplicação ou regras escritas em um domínio ou ambiente de execução específico. Cada atividade pode suportar ganchos de pré-interceptação e ganchos de pós-interceptação na execução no código do usuário. Cada atividade tem semântica e comportamento de execução em tempo de execução associados (por exemplo, gerenciamento de estado, transações, tratamento de evento e tratamento de exceção). Atividades podem compartilhar estado ou recursos com outras atividades.

A figura 4 é um diagrama que ilustra um sistema 400 para processar atividades de

fluxo de trabalho de acordo com uma modalidade da invenção. O sistema 400 inclui um processador 402 que pode ser uma unidade de processamento ou uma coleção de unidades de processamento. O sistema 400 também inclui uma área de memória 404 para armazenar dados acessíveis pelo processador 402. Na modalidade, o sistema 400 pode ser um computador com um ou mais processadores ou unidades de processamento (por exemplo, processador 402) e uma memória de sistema (por exemplo, área de memória 404) e com pelo menos um outro componente conhecido pelos versados na técnica que acopla vários componentes de sistema, que inclui a memória do sistema, no processador 402.

Em um exemplo, a assinante de memória 404 pode incluir mídia legível por computador, seja mídia volátil, não volátil, removível ou não removível, implementada em qualquer método ou tecnologia para armazenamento de informação, tais como instruções legíveis por computador, estruturas de dados, módulos de programa e outros dados. Por exemplo, mídia de armazenamento no computador inclui RAM, ROM, EEPROM, memória flash ou outra tecnologia de memória, CD-ROM, discos versáteis digitais (DVD) ou outro armazenamento em disco ótico, cassetes magnéticos, fita magnética, armazenamento em disco magnético ou outro dispositivo de armazenamento magnético, ou qualquer outra mídia que pode ser usada para armazenar a informação desejada e que pode ser acessada pelo sistema 400. A memória 404 também pode incluir mídia de comunicação que incorpora instruções legíveis por computador, estruturas de dados, módulos de programa ou outros dados em um sinal de dados modulado, tais como uma onda portadora ou outro mecanismo de transporte, e inclui qualquer mídia de distribuição de informação. Versados na técnica estão familiarizados com o sinal de dados modulado, que tem uma ou mais de suas características ajustadas ou modificadas de uma maneira tal para codificar informação no sinal. Mídia com fios, tais como rede com fios ou conexão direta com fios, e mídia sem fios, tais como acústica, RF, infravermelho, e outras mídias sem fios, são exemplos de mídia de comunicação. Combinações de qualquer um dos expostos também podem ser incluídas no escopo da mídia legível por computador.

No exemplo, a área de memória 404 armazena uma pluralidade de atividades 406 para processamento em um fluxo de trabalho (por exemplo, o fluxo de trabalho 300). Cada uma da pluralidade de atividades 406 inclui um ou mais itens de trabalho, e os itens de trabalho podem ser organizados em uma estrutura hierárquica, tal como uma estrutura de árvore (veja figura 5). No processamento da pluralidade de atividades 406, o processador 402 acessa ou executa um agendador 408, que é configurado para ajustar um conjunto organizado de atividades.

Por exemplo, o processador 408 acessa os itens de trabalho na pluralidade de atividades 406 por meio de um componente ou de um conjunto de instruções legíveis por computador, tal como o agendador 408, para enfileirar os itens de trabalho 422 em uma fila

410. Um expedidor 412, acessível pelo processador 402, expede os itens de trabalho 422 para execução. Por exemplo, um item de trabalho 422-1 pode incluir um método de atividade 424, rotina ou uma coleção de códigos para realizar uma função de "solicitar entrada de um usuário". Um ou mais outros métodos de atividade, rotinas ou códigos podem ser incluídos em cada um dos itens de trabalho 422 sem fugir do escopo da invenção.

Uma vez que os itens de trabalho 422 são expedidos pelo expedidor 412, o processador 402 executa cada um dos métodos 424 nos itens de trabalho 422 em 414. No exemplo do item de trabalho 422-1, o processador 402 pode cuidar para que um usuário, por meio de uma interface de usuário (UI), insira a informação ou dados solicitados. Em uma outra modalidade, o processador 402 pode conectar ou acessar uma fonte de dados externa para solicitar entrada do usuário. Mediante a conclusão do método da atividade 424, o processador 402 conclui a execução dos itens de trabalho 422 em 416. Em uma modalidade, o processador 402 passiva o estado de execução dos itens de trabalho em 418 em um armazenamento de dados 420.

Em uma outra modalidade, o processador 402 executa os itens de trabalho 422 de acordo com um autômato de estado, tal como o autômato mostrado na figura 6, que é um diagrama que ilustra um autômato de estado exemplar 600 que descreve os estados de processamento dos itens de trabalho associados com uma atividade de acordo com uma modalidade da invenção. Em um exemplo, o autômato de estado 600 pode incluir um estado inicializado, um estado de execução e um estado encerrado (da forma mostrada na figura 4). Em uma outra modalidade, o autômato de estado 600 inclui um estado inicializado 602, um estado de execução 604, um estado de cancelamento 606, um estado defeituoso 608, um estado de compensação 610 e um estado encerrado 612.

Por exemplo, o autômato de estado 600 descreve um fluxo de processo de execução dos itens de trabalho (por exemplo, itens de trabalho 422) em uma atividade de fluxo de trabalho. O item de trabalho 422-1, da forma ilustrada na figura 4, é inicializado primeiro quando ele está enfileirado na fila 410. A seguir, o item de trabalho 422-1 é retirado da fila até o expedidor 412 antes de ser executado no estado de execução (por exemplo, estado de execução 604, na figura 6). Dependendo dos parâmetros ou condições durante a execução do item de trabalho 422-1, o item de trabalho 422-1 pode prosseguir até o estado de cancelamento 606 do estado defeituoso 608. Em uma modalidade, o item de trabalho 422-1 pode prosseguir do estado de cancelamento 606 até o estado defeituoso 608. Em uma modalidade alternativa, o estado de compensação 610 descreve um conjunto de operações ou funções a ser realizadas quando o defeito ou exceção tiver ocorrido. Por exemplo, suponha que uma exceção ocorra durante a execução do item de trabalho (por exemplo, item de trabalho 422-1), tal como a falta de um parâmetro para uma função. O sistema 400 transiciona o item de trabalho 422-1 para o estado defeituoso 608. Fazendo isto, o sistema 400 também reali-

za a operação da coleta de lixo (por exemplo, remoção da parte previamente executada das operações do cache ou da memória, reinício dos valores de parâmetro, ou congêneres) no estado de compensação 610 antes de transicionar o item de trabalho 422-1 para o estado encerrado 612.

Em uma modalidade, programas projetados de acordo com a estrutura de fluxo de trabalho que incorpora aspectos da invenção podem ser visitados por qualquer número de linhas de execução dos níveis inferiores (por exemplo, um ambiente de execução gerenciado, tais como um tempo de execução em linguagem comum (CLR) ou nível OS). Em uma outra modalidade, um agendador (por exemplo, agendador 408) pode usar uma linha de execução CLR dedicada para uma dada realização ou execução de atividades.

Além do mais, um indicador de execução para uma atividade correspondente a uma atividade pode ser visualizado como uma linha de execução sob a estrutura de fluxo de trabalho (WF) que incorpora aspectos da invenção. Como tal, linhas de execução WF intercalam em pontos de espera ou mediante agendamento explícito fora de ordem de uma maneira assíncrona pela atividade composta pai.

Agora, em relação à figura 7A, um diagrama de blocos ilustra uma execução síncrona dos itens de trabalho em uma atividade em um fluxo de trabalho de acordo com uma modalidade da invenção. Em uma modalidade, a execução síncrona dos itens de trabalho habilita uma execução intercalada de linhas de execução associadas com o fluxo de trabalho. Em uma modalidade, o fluxo de trabalho define uma SynchronizationScopeActivity para sincronizar o acesso ao estado compartilhado em um exemplo de uma atividade ilustrada no Apêndice A. Por exemplo, suponha que duas atividades no fluxo de trabalho desejam acessar um diretório em um armazenamento de dados para realizar duas operações diferentes. Neste exemplo, a $Activity_1$ pode desejar acessar arquivos no diretório antes de atualizar informação nos arquivos lidos. Ao mesmo tempo, a $Activity_2$ pode desejar modificar o mesmo conjunto de arquivos acessados pela $Activity_1$. Como tal, acessos a recursos compartilhados pela $Activity_1$ e pela $Activity_2$ precisam ser sincronizados e gerenciados. O Apêndice A ilustra um conjunto de operações SynchronizationScope que mostra uma implementação exemplar da sincronia ou intercalação do estado compartilhado através das linhas de execução de uma maneira declarativa.

Tipicamente, sistemas existentes realizam tal sincronia contando com travas de acesso ou outros métodos fornecidos por um sistema operacional (OS). Tais travas do OS, embora forneçam funcionalidades básicas para alcançar o propósito desejado, são embutidas no código, inflexíveis e inadequadas para a fundação ou estrutura de fluxo de trabalho extensível. Além do mais, comumente, linhas de execução no nível do OS incluem chaves de contexto associadas com as travas do OS (por exemplo, endereços de memória gravados ou identificados, endereços de memória anteriores, alocações de pilha ou congêneres).

Além do mais, as travas do OS não sobrevivem ao processo de passivação em virtude de todos os apontadores, pilhas, etc., associados com as travas do OS deixarem de restaurar os valores previamente atribuídos nas travas antes da passivação. Além do mais, embora fluxos de trabalho possam executar em diferentes máquinas durante seu ciclo de vida, travas OS são válidas somente para a máquina em que elas foram criadas.

Modalidades da invenção atribuem indicadores de sincronia 718 para itens de trabalho na atividade para garantir que linhas de execução acessem de forma síncrona os recursos compartilhados sem conflito ou impasse. Aspectos da invenção habilitam desenvolvedores a projetar linhas de execução virtuais e leves no topo das linhas de execução de um ambiente de execução físico ou gerenciado. Em uma outra modalidade, as linhas de execução são leves para não se anexar ou se associar a nenhuma referência física ou de hardware. Como tal, as linhas de execução sobrevivem a ciclos de passivação ou podem ser preservadas depois de ser armazenadas em um armazenamento de dados.

Por exemplo, suponha que um programa inclui uma função de chamada de volta e, durante a execução da função de chamada de volta, determina-se que o programa precisa ser passivado em um armazenamento de dados. Linguagens de programação e de modelo existentes exigem que todos os parâmetros associados com o estado de execução do programa sejam salvos como um objeto. O estado do programa pode ser restaurado em um ponto posterior no tempo a partir do objeto. Entretanto, usualmente, o objeto inclui linhas de execução que têm associação embutida no código com contexto de hardware ou ajustes em que o programa é executado. Modalidades da invenção superam esta dependência.

Novamente, em relação à figura 7A, um diagrama ilustra a execução síncrona das linhas de execução quando mais de uma linha de execução tentar acessar o recurso compartilhado. Inicialmente, uma coleção 902 dos itens de trabalho em uma atividade é agendada para ser processada. Em uma modalidade, os itens de trabalho na atividade podem ser organizados em uma estrutura de árvore, tal como a atividade 502 mostrada na figura 5. Um agendador 704 acessa a coleção 702 dos itens de trabalho e coloca os itens de trabalho em fila em uma fila do agendador 706 para ser processados. Em uma modalidade, a rotina de tratamento de sincronia 718 é atribuída para cada uma das atividades na fila do agendador 906, indicando que um recurso compartilhado em particular deve ser acessado pela atividade. Por exemplo, um item de trabalho 708-2 e um item de trabalho 708-N acessam um recurso compartilhado (por exemplo, um local de memória). Como tal, ambos os itens de trabalho 708-2 e 708-N incluem indicadores de sincronia 718. Por outro lado, um item de trabalho 708-3, que não acessa um recurso compartilhado com outros itens de trabalho, não inclui uma rotina de tratamento de sincronia 718, e aquele recurso compartilhado em particular que está sendo compartilhado por um ou mais dos itens de trabalho.

Em uma modalidade, a rotina de tratamento de sincronia 718 é uma seqüência de

caracteres com sinal parecida com um objeto de exclusão mútua nomeado ("mutex"). Em uma modalidade, um filtro de sincronia 720 é aplicado nos itens de trabalho com a rotina de tratamento de sincronia 718 atribuída.

Uma vez que a rotina de tratamento de sincronia 718 é atribuída, um valor com sinal é computado para cada um dos itens de trabalho com base na rotina de tratamento de sincronia 718 atribuída. Nas modalidades em que os itens de trabalho são parte de uma estrutura hierárquica (por exemplo, uma estrutura de árvore) na atividade, o valor com sinal é computado com base na rotina de tratamento de sincronia 718 atribuída e em um local do item de trabalho na hierarquia da atividade. Em uma modalidade, um componente com sinal (da forma mostrada na figura 9) ou um gerente de trava monitora ou gerencia a rotina de tratamento de sincronia 718 e os recursos compartilhados para cada um dos itens de trabalho na árvore de atividade.

Os itens de trabalho com os valores com sinal computados são ordenados a seguir em uma fila de sincronia 712 com base no valor com sinal computado associado com cada um dos itens de trabalho. Em uma modalidade, uma função AcquireLock pode ser usada na ordenação e na determinação se os acessos ao recurso compartilhado forem permitidos. Por exemplo, da forma ilustrada na figura 7A, em virtude de os valores com sinal computados, a função AcquireLock ordena o item de trabalho 708-2 em uma posição precedente ao item de trabalho 708-N na fila de sincronia 712.

Em uma modalidade, a função AcquireLock realiza a ordenação dos itens de trabalho na fila de sincronia 712, primeiro, pela coleta de todos os indicadores de sincronia 718 que pertencem à mesma atividade. Como tal, impasses não ocorrem. A figura 7B é um diagrama que ilustra a ordenação da fila de sincronia 712 pela função AcquireLock de acordo com uma modalidade da invenção. Por exemplo, uma árvore de atividade simplista 732 inclui um nó raiz / pai 722 e dois nós filhos 724 e 726, e cada qual inclui um work $item_1$ e um work $item_2$, respectivamente.

Como exposto, alguns dos itens de trabalho na árvore de atividade podem não incluir uma rotina de tratamento de sincronia. Como tal, se algum item de trabalho no nó filho / folha da árvore de atividade não tiver uma rotina de tratamento de sincronia, a função AcquireLock não prosseguirá adicionalmente na estrutura hierárquica da árvore de atividade. Em uma modalidade, durante a ordenação, a função AcquireLock também remove itens de trabalho duplicados para evitar qualquer impasse na fila de sincronia 712.

Continuando a atravessar os nós da árvore de atividade, a seguir, a função AcquireLock tenta identificar o item de trabalho no nó raiz ou pai da árvore de atividade, tal como o nó raiz 722. Depois de encontrar o nó raiz ou pai, a função AcquireLock determina se uma lista ou dicionário de indicadores coletados para o nó raiz ou pai 722 inclui todos os indicadores de sincronia 718 para todos os seus filhos.

Na figura 7B, uma lista de indicadores coletados 728 inclui informação, tais como "H" (denotando indicadores de sincronia) e "GL" (denotando Travas Concedidas). Da forma ilustrada, a lista 728 indica que a rotina de tratamento de sincronia 718 atribuída ao nó filho 724 foi coletada e que o nó raiz 722 é o titular da rotina de tratamento de sincronia 718 (isto é, permissão de acessar um recurso compartilhado em particular) para o work $item_1$ do nó filho 724.

Por outro lado, a lista 728 também indica que o nó raiz 722 tem a rotina de tratamento de sincronia 718 para o nó filho 726, mas o nó raiz 722 não é o titular da rotina de tratamento de sincronia 718 (isto é, o nó raiz 722 não tem acesso ao recurso compartilhado). Como tal, o work $item_2$ do nó filho 726 é adicionado em uma lista de espera 730 e a função AcquireLock realizará uma outra iteração do processo para garantir que o nó raiz 722 obtenha a rotina de tratamento de sincronia 718 para o nó filho 726.

Uma vez que a função AcquireLock ordena a fila de sincronia 712 de acordo com a descrição exposta, um expedidor 710 expede os itens de trabalho (por exemplo, 708-2) na fila de sincronia 712 a ser executados no estado de execução no qual métodos de atividade ou funções nos itens de trabalho são processados. Como tal, os itens de trabalho são seqüencialmente executados a partir da fila de sincronia para tornar serial o acesso ao recurso compartilhado em particular e efetuar uma execução síncrona das linhas de execução associadas com os itens de trabalho.

Agora, em relação à figura 8, um fluxograma que ilustra um método para executar de forma síncrona atividades acessa um recurso compartilhado em particular de acordo com uma modalidade da invenção. Em um exemplo, o método ilustrado na figura 8 pode ser realizado por componentes executáveis por computador incluídos em uma mídia legível por computador 900 ilustrada na figura 9. Por exemplo, um componente de armazenamento 902 armazena uma pluralidade de itens de trabalho em uma fila (por exemplo, uma fila do agendador) ou coloca em fila a pluralidade de itens de trabalho para a execução em 802. O um ou mais da pluralidade de itens de trabalho é associado com uma atividade em um fluxo de trabalho e com um ou mais da pluralidade de itens de trabalho que é organizado em uma estrutura de árvore ou em outra estrutura hierárquica na atividade. Cada um dos itens de trabalho é associado com uma linha de execução.

Em 804, um componente de sincronia 904 atribui uma rotina de tratamento de sincronia para cada um da pluralidade de itens de trabalho na fila. A rotina de tratamento de sincronia indica o recurso compartilhado em particular a ser acessado pela pluralidade de itens de trabalho. Em 806, um componente com sinal 906 computa um valor com sinal para cada um dos itens de trabalho com base na rotina de tratamento de sincronia atribuída e em um local dos itens de trabalho na estrutura de árvore na atividade. Um componente de ordenação 908 ordena os itens de trabalho em uma fila de sincronia com base no valor com sinal

associado com cada um dos itens de trabalho em 808. Em 810, um componente de execução 910 executa cada um dos itens de trabalho ordenados na fila de sincronia para tornar serial o acesso ao recurso compartilhado em particular e efetuar uma execução síncrona das linhas de execução associadas com o fluxo de trabalho.

Em uma modalidade, a mídia legível por computador 900 inclui adicionalmente um componente de definição 912 para definir declarativamente a rotina de tratamento de sincronia atribuída para cada um dos itens de trabalho pela exposição das propriedades da rotina de tratamento de sincronia para cada um dos itens de trabalho. Em uma ainda outra modalidade alternativa, a mídia legível por computador 900 inclui um componente de passivação 914 para passivar a fila de sincronia com os itens de trabalho e os valores com sinal associados em um armazenamento de dados.

Embora descrito em conjunto com um ambiente de sistema de computação exemplar, tal como o sistema 400 da figura 4, modalidades da invenção são operacionais com inúmeros outros ambientes ou configurações de sistema de computação de uso geral ou de uso especial. Não pretende-se que o ambiente do sistema de computação sugira nenhuma limitação ao escopo do uso ou à funcionalidade de nenhum aspecto da invenção. Além do mais, o ambiente do sistema de computação não deve ser interpretado com qualquer dependência ou exigência relacionada a nenhum dos componentes ou de suas combinações ilustrados no ambiente operacional exemplar. Exemplos de sistemas, ambientes e/ou configurações de computação bem conhecidos que podem ser adequados para uso com aspectos da invenção incluem, mas sem limitações, computadores pessoais, computadores servidores, dispositivos de mão ou portáteis, sistemas multiprocessadores, sistemas com base em microprocessador, conversores de sinal de freqüência, aparelhos eletrônicos programáveis pelo cliente, telefones celulares, PCs em rede, minicomputadores, computadores de grande porte, ambientes de computação distribuída que incluem qualquer um dos sistemas ou dispositivos expostos, e congêneres.

Modalidades da invenção podem ser descritas no contexto geral das instruções executáveis por computador, tais como módulos de programa, executadas por um ou mais computadores ou outros dispositivos. No geral, módulos de programa incluem, mas sem limitações, rotinas, programas, objetos, componentes e estruturas de dados que realizam tarefas em particular ou implementam tipos de dados abstratos em particular. Aspectos da invenção também podem ser praticados em ambientes de computação distribuída em que tarefas são realizadas por dispositivos de processamento remotos que são ligados por meio de uma rede de comunicações. Em um ambiente de computação distribuída, módulos de programa podem ficar localizados em mídia de armazenamento no computador tanto local quanto remota, incluindo dispositivo de armazenamento em memória.

Em operação, o sistema 400 executa instruções executáveis por computador, tais

como aquelas ilustradas nas figuras, tal como na figura 7, para implementar aspectos da invenção.

A ordem de execução ou de desempenho das operações nas modalidades da invenção aqui ilustrada e descrita não é essencial, a menos que de outra forma especificado. Isto é, as operações podem ser realizadas em qualquer ordem, a menos que de outra forma especificado, e modalidades da invenção podem incluir operações adicionais ou menos operações do que aquelas aqui divulgadas. Por exemplo, percebe-se que a execução ou o desempenho de uma operação em particular antes, durante ou depois de uma outra operação está no escopo dos aspectos da invenção.

Modalidades da invenção podem ser implementadas com instruções executáveis por computador. As instruções executáveis por computador podem ser organizadas em um ou mais componentes ou módulos executáveis por computador. Aspectos da invenção podem ser implementados com qualquer número e organização de tais componentes ou módulos. Por exemplo, aspectos da invenção não são limitados às instruções executáveis por computador ou aos componentes ou módulos específicos ilustrados nas figuras e aqui descritos. Outras modalidades da invenção podem incluir diferentes instruções executáveis por computador ou componentes com mais ou menos funcionalidades do que aqui ilustrado e descrito.

Durante a introdução dos elementos dos aspectos da invenção ou das suas modalidades, pretende-se que os artigos “um”, “uma”, “o”, “a”, “dito” e “dita” signifiquem que há um ou mais dos elementos. Pretende-se que os termos “compreendendo”, “incluindo” e “tendo” sejam inclusivos e signifiquem que pode haver elementos adicionais diferentes dos elementos listados.

Tendo sido descritos aspectos da invenção com detalhes, ficará aparente que modificações e variações são possíveis sem fugir do escopo dos aspectos da invenção definido nas reivindicações anexas. Já que várias mudanças podem ser feitas nas construções, produtos e métodos expostos sem fugir do escopo dos aspectos da invenção, pretende-se que todo o assunto contido na descrição exposta e mostrado nos desenhos anexos seja interpretado como ilustrativo e não em um sentido limitante.

APÊNDICE A

```
<myActivities:Parallel x:Name="pxxx"
xmlns:myActivities="http://schemas.com/myActivities"
xmlns:x="http://schemas. microsoft.com/winfx/2006/xaml"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/workflow">
<SynchronizationScopeActivity x:Name="sl " SynchronizationHandles="a">
<myActivities:Parallel x:Name="pl ">
<SynchronizationScopeActivity x:Name="s2" SynchronizationHandles="a">
```

```
<myActivities:WriteLine x:Name="w3" Text="One'7>
<myActivities:WriteLine x:Name="w4" Text="Two"/>
</SynchronizationScopeActivity>
<SynchronizationScopeActivity x:Name="s3" SynchronizationHandles="b">
<myActivities:WriteLine x:Name="w5" Text="Three"/>
<myActivities:WriteLine x:Name="w6" Text="Four'7>
</SynchronizationScopeActivity>
</myActivities:Parallel>
</SynchronizationScopeActivity>
<SynchronizationScopeActivity x:Name="s4" SynchronizationHandles="b">
<myActivities:Paraliel x:Name="p2">
<SynchronizationScopeActivity x:Name="s5" SynchronizationHandles="b">
<myActivities:WriteLine x:Name="w9" Text="Five"/>
<myActivities:WriteLine x:Name="wlO" Text="Six"/>
</SynchronizationScopeActivity>
<SynchronizationScopeActivity x:Name="s6" SynchronizationHandles="a">
 <myActivities:WriteLine x:Name="wl 1" Text="Seven"/>
<myActivities:WriteLine x:Name="wl2" Text="Eight"/>
</SynchronizationScopeActivity>
</myActivities:Parallel>
</SynchronizationScopeActivity>
</myActivities:Parallel>
```

APÊNDICE B

Em um ainda outro exemplo, o seguinte ilustra uma seqüência exemplar da implementação da execução de linha de execução assíncrona de acordo com uma modalidade da invenção.

1) Filtro de Sincronia é aplicado em todas as atividades que têm o atributo [SupportsSynchronization].

2) RootActivity e todos os escopos de sincronia mantêm um Dicionário de {indicador, GrantedLock}. Cada estrutura GrantedLock} mantém a atividade e a WaitList. GrantedLock pode ser serializado assim como o Dicionário de maneira tal que ambos possam sobreviver à passivação.

3) Cada escopo de sincronia é um gerente de trava para seus escopos de sincronia filhos. A atividade raiz é o gerente de trava padrão. Um gerente de trava é responsável por conceder as travas para o escopo de sincronia filho e também por manter uma lista de espera dos escopos de sincronia filhos que podem não receber travas.

4) Há uma hierarquia das linhas de execução WF (por exemplo, com base na hie-

rarquia da atividade) e, similarmente, uma hierarquia dos gerentes de trava / escopos de sincronia.

5) Em seu método de execução, SynchronizationFilter chama AcquireLocks passando a si mesmo como uma chamada de volta.

6) Se AcquireLock retornar verdadeiro, a execução da atividade prossegue. Caso contrário, ela permanece no estado de execução.

7) Na AcquireLock:

a. A atividade coleta os indicadores de sincronia e *todos* os seus filhos. Se nenhum dos filhos tiver tais indicadores de sincronia, pára-se de ir além na hierarquia. Então, ela remove duplicatas e ordena todos os indicadores. Isto é para evitar todos os impasses.

b. Se nenhum indicador for encontrado, AcquireLock retorna verdadeiro. A execução do escopo de sincronia pode prosseguir desde que ele não esteja sincronizando nada (no-op).

c. AcquireLock continua caminhando pela hierarquia pai procurando um filho que esteja no escopo de sincronia ou até que ela encontre a raiz. Em cada etapa, quando ela encontra um pai de escopo de sincronia, para cada um dos indicadores que ela tiver coletado, ela verifica se o dicionário de travas concedidas do pai tem o indicador. Se não, ela se adiciona como a atividade para o dado indicador para o qual a trava foi concedida. Se ele já tiver o indicador e ele não for o titular, ele se adiciona na lista de espera. Em qualquer ponto que ele estiver adicionado a pelo menos uma das listas de espera, AcquireLock retorna falso. Se nenhum pai de escopo de sincronia tiver lista de indicadores não vazia, interrompe-se e retorna-se verdadeiro, desde que o pai já tenha adquirido todas as travas. Lembre que o filtro de sincronia do pai sempre é executado antes do filho.

8) O filtro de sincronia se inscreve para a atividade para o estado fechado. Uma vez que ele prossegue até o estado fechado, o filtro chama ReleaseLock.

9) Na ReleaseLock:

a. Caminha na cadeia do pai, para cada etapa, para cada indicador, coleta as atividades que estão esperando em um dado indicador.

b. Para todas as atividades em espera, tenta readquirir as travas. Para aquelas para as quais pode adquirir as travas, invoca a chamada de volta do filtro.

REIVINDICAÇÕES

1. Sistema (400) para implementar sincronia entre linhas de execução em um fluxo de trabalho, **CARACTERIZADO** pelo fato de que o dito sistema (400) compreende:

uma área de memória (404) para armazenar uma pluralidade de itens de trabalho (422) em uma fila do agendador (410), os ditos um ou mais da pluralidade de itens de trabalho (422) sendo organizados hierarquicamente (500) em uma atividade (406) e sendo associados com a atividade (406) em um fluxo de trabalho, cada um dos itens de trabalho (422) sendo associado com uma linha de execução;

um processador (402) configurado para executar instruções executáveis por computador para:

atribuir uma rotina de tratamento de sincronia (718) a cada um dos itens de trabalho (422) na fila do agendador (410), a dita rotina de tratamento de sincronia (718) indicando um recurso compartilhado em particular a ser acessado pelo item de trabalho (422), o dito recurso compartilhado em particular sendo compartilhado por um ou mais dos itens de trabalho (422);

para cada um dos itens de trabalho (422), computar um valor com sinal com base na rotina de tratamento de sincronia atribuída (718) e em um local do item de trabalho na hierarquia (500) na atividade (406);

armazenar os itens de trabalho em uma fila de sincronia (712) com base no valor com sinal computado com cada um dos itens de trabalho (422); e

executar seqüencialmente cada um dos itens de trabalho (422) ordenados na fila de sincronia (712) para tornar serial o acesso ao recurso compartilhado em particular e efetuar uma execução síncrona das linhas de execução associadas com os itens de trabalho (422).

2. Sistema (400), de acordo com a reivindicação 1, **CARACTERIZADO** pelo fato de que o processador (402) é adicionalmente configurado para definir declarativamente a rotina de tratamento de sincronia (718) atribuída para cada um dos itens de trabalho (422) pela exposição das propriedades da rotina de tratamento de sincronia (718) para cada um dos itens de trabalho (422).

3. Sistema (400), de acordo com a reivindicação 1, **CARACTERIZADO** pelo fato de que a rotina de tratamento de sincronia (718) atribuída de cada um dos itens de trabalho (422) define um escopo de linha de execução associado com cada um dos itens de trabalho (422).

4. Sistema (400), de acordo com a reivindicação 1, **CARACTERIZADO** pelo fato de que a hierarquia (500) na atividade (406) é uma estrutura de árvore, e em que o processador (402) é configurado para executar instruções executáveis por computador para ordenar os itens de trabalho (422) com base no local na estrutura de árvore.

5. Sistema (400), de acordo com a reivindicação 1, **CARACTERIZADO** pelo fato de

que o processador (402) é adicionalmente configurado para passivar a fila de sincronia (712) armazenando os itens de trabalho (422) e os valores com sinal associados em um armazenamento de dados (420).

6. Sistema (400), de acordo com a reivindicação 1, **CARACTERIZADO** pelo fato de que o processador (402) é configurado para executar os itens de trabalho (422) pela transição dos itens de trabalho (422) a um estado de execução (604) de acordo com um autômato de estado (600) associado com o processamento do fluxo de trabalho.

7. Método para a execução sincronizada de atividades em um fluxo de trabalho acessando um recurso compartilhado em particular, **CARACTERIZADO** pelo fato de que o dito método compreende:

enfileirar uma pluralidade de itens de trabalho (422) para execução, os ditos um ou mais da pluralidade de itens de trabalho sendo organizados em uma seqüência em uma atividade (406) e sendo associados com a atividade (406) em um fluxo de trabalho, os ditos cada um dos itens de trabalho (422) sendo uma linha de execução;

atribuir uma rotina de tratamento de sincronia (718) para cada um da pluralidade de itens de trabalho (422), a dita rotina de tratamento de sincronia (718) indicando o recurso compartilhado em particular a ser acessado pela pluralidade de itens de trabalho (422), o dito recurso compartilhado em particular sendo compartilhado por um ou mais dos itens de trabalho (422);

para cada um dos itens de trabalho (422), computar um valor com sinal com base na rotina de tratamento de sincronia atribuída (718) e em um local dos itens de trabalho na seqüência na atividade (406);

ordenar os itens de trabalho (422) em uma fila de sincronia (712) com base no valor com sinal computado associado com cada um dos itens de trabalho (422); e

executar seqüencialmente cada um dos itens de trabalho (422) ordenados na fila de sincronia (712) para tornar serial o acesso ao recurso compartilhado em particular e efetuar uma execução síncrona das linhas de execução associadas com os itens de trabalho (422).

8. Método, de acordo com a reivindicação 7, **CARACTERIZADO** pelo fato de que compreende adicionalmente definir declarativamente a rotina de tratamento de sincronia atribuída (718) para cada um dos itens de trabalho pela exposição das propriedades da rotina de tratamento de sincronia (718) para cada um dos itens de trabalho (422).

9. Método, de acordo com a reivindicação 7, **CARACTERIZADO** pelo fato de que atribuir a rotina de tratamento de sincronia (718) compreende atribuir uma seqüência de caracteres com sinal.

10. Método, de acordo com a reivindicação 7, **CARACTERIZADO** pelo fato de que atribuir a rotina de tratamento de sincronia (718) compreende definir um escopo de linha de execução associado com cada um dos itens de trabalho (422).

11. Método, de acordo com a reivindicação 7, **CARACTERIZADO** pelo fato de que a seqüência na atividade (406) corresponde a uma seqüência para atravessar a árvore de estrutura (500), e em que a ordenação compreende arranjar os itens de trabalho (422) com base no local na árvore de estrutura.

12. Método, de acordo com a reivindicação 7, **CARACTERIZADO** pelo fato de que compreende adicionalmente passivar a fila de sincronia (712) com os itens de trabalho (422) e os valores com sinal associados em um armazenamento de dados (420).

13. Método, de acordo com a reivindicação 7, **CARACTERIZADO** pelo fato de que a execução compreende executar os itens de trabalho pela transição a um estado de execução de acordo com um autômato de estado associado com o processamento do fluxo de trabalho.

14. Método, de acordo com a reivindicação 7, **CARACTERIZADO** pelo fato de que uma ou mais mídias legíveis por computador têm instruções executáveis por computador para realizar o método da reivindicação 1.

15. Mídia legível por computador (900) com componentes executáveis por computador para executar atividades de forma síncrona, **CARACTERIZADA** pelo fato de que os ditos componentes executáveis por computador compreendem:

um componente de armazenamento (902) para armazenar uma pluralidade de itens de trabalho (422) em uma fila (410), o dito um ou mais da pluralidade de itens de trabalho (422) sendo associado com uma atividade (406) em um fluxo de trabalho, o dito um ou mais da pluralidade de itens de trabalho (422) sendo organizado em uma estrutura de árvore (500) na atividade (406), os ditos cada um dos itens de trabalho (422) sendo associados com uma linha de execução;

um componente de sincronia (904) para atribuir uma rotina de tratamento de sincronia (718) a cada um da pluralidade de itens de trabalho (422) na fila (410), a dita rotina de tratamento de sincronia (718) indicando o recurso compartilhado em particular a ser acessado pela pluralidade de itens de trabalho (422), o dito recurso compartilhado em particular sendo compartilhado por um ou mais dos itens de trabalho (422);

um componente com sinal (906) para computar um valor com sinal para cada um dos itens de trabalho (422) com base na rotina de tratamento de sincronia atribuída (718) e em um local dos itens de trabalho (422) na estrutura de árvore (500) na atividade (406);

um componente de ordenação (908) para ordenar os itens de trabalho (422) em uma fila síncrona (712) com base no valor com sinal associado com cada um dos itens de trabalho (422); e

um componente de execução (910) para executar cada um dos itens de trabalho (422) na fila de sincronia (712) para tornar serial o acesso ao recurso compartilhado em particular e efetuar uma execução síncrona das linhas de execução associadas com o fluxo de

trabalho.

16. Mídia legível por computador (900), de acordo com a reivindicação 15, **CARACTERIZADA** pelo fato de que compreende adicionalmente um componente de definição (914) para definir declarativamente a rotina de tratamento de sincronia (718) atribuída para cada um dos itens de trabalho (422) pela exposição das propriedades da rotina de tratamento de sincronia para cada um dos itens de trabalho (422).

17. Mídia legível por computador (900), de acordo com a reivindicação 15, **CARACTERIZADA** pelo fato de que o componente de sincronia (904) compreende definir um escopo de linha de execução associado com cada um dos itens de trabalho (422).

18. Mídia legível por computador (900), de acordo com a reivindicação 15, **CARACTERIZADA** pelo fato de que compreende adicionalmente um componente de passivação (912) para passivar a fila síncrona (712) com os itens de trabalho (422) e os valores com sinal associados em um armazenamento de dados (420).

19. Mídia legível por computador (900), de acordo com a reivindicação 18, **CARACTERIZADA** pelo fato de que o componente de passivação (912) armazena a fila de sincronia (712) com os itens de trabalho (422), o sinal associado e o estado de execução no armazenamento de dados (420).

20. Mídia legível por computador (900), de acordo com a reivindicação 15, **CARACTERIZADA** pelo fato de que o componente de execução (910) compreende executar os itens de trabalho (422) pela transição a um estado de execução (604) de acordo com um autômato de estado (600) associado com o processamento do fluxo de trabalho.

FIG. 1

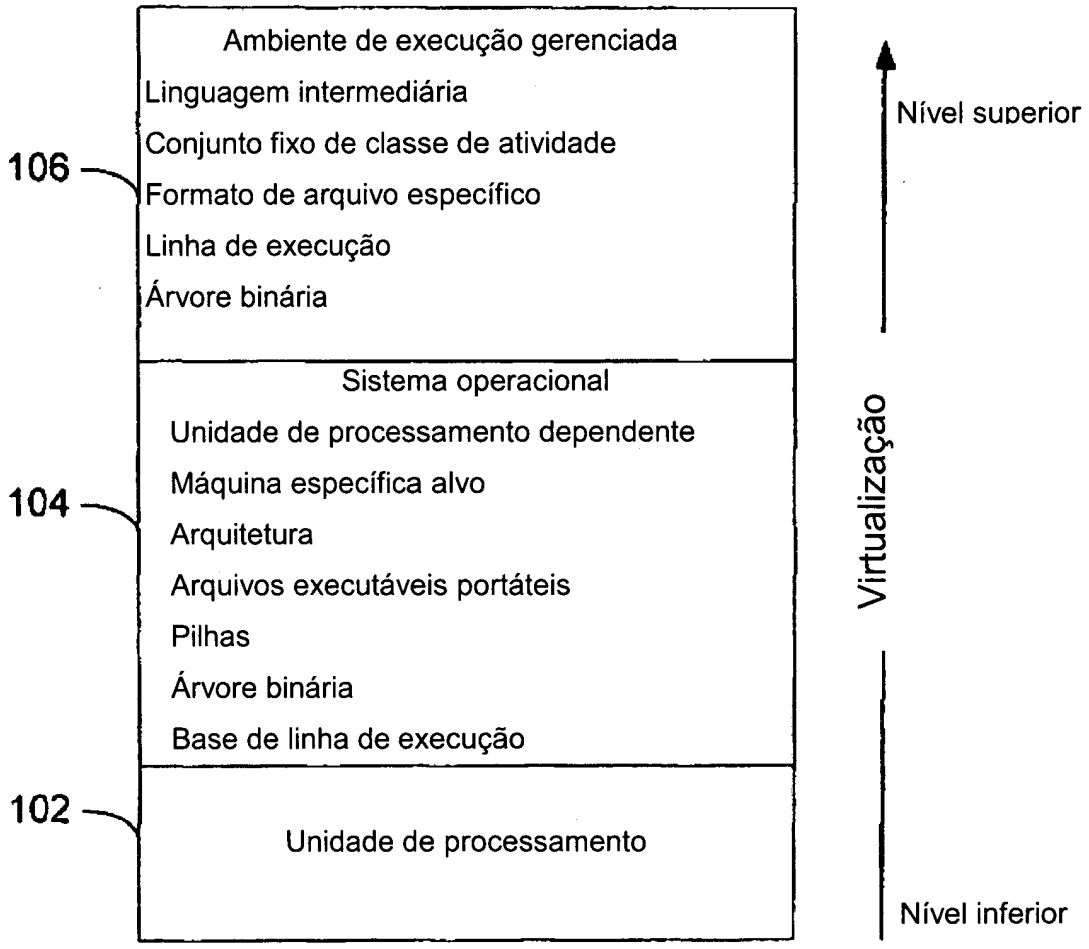

Ambiente de execução gerenciada
Linguagem intermediária
Conjunto fixo de classe de atividade
106
Formato de arquivo específico
Linha de execução
Árvore binária
Sistema operacional
Unidade de processamento dependente
Máquina específica alvo
104
Arquitetura
Arquivos executáveis portáteis
Pilhas
Árvore binária
Base de linha de execução
102
Unidade de processamento
Nível superior
Virtualização
Nível inferior

FIG. 2

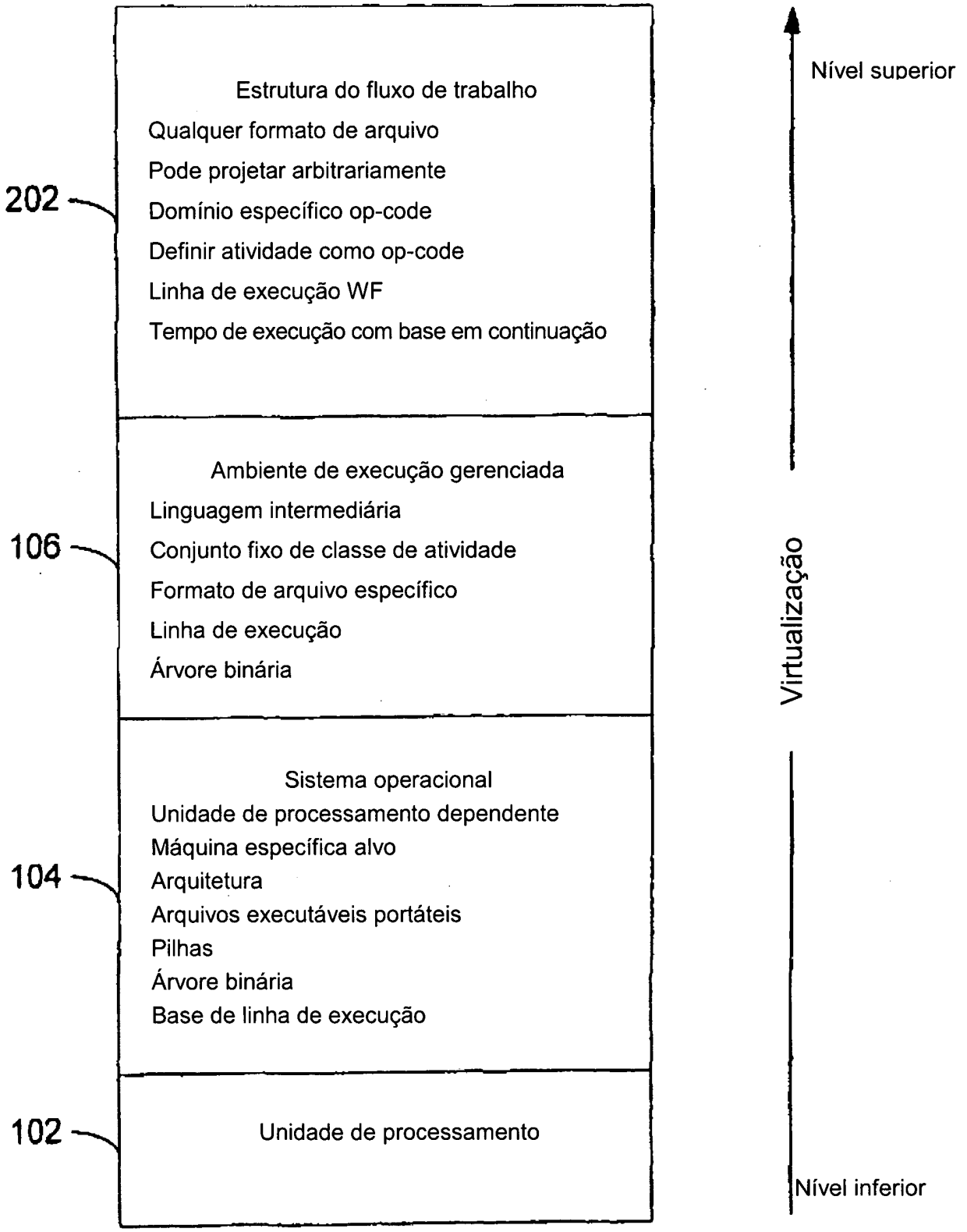

Estrutura do fluxo de trabalho
Qualquer formato de arquivo
Pode projetar arbitrariamente
202
Domínio específico op-code
Definir atividade como op-code
Linha de execução WF
Tempo de execução com base em continuação
Ambiente de execução gerenciada
Linguagem intermediária
106
Conjunto fixo de classe de atividade
Formato de arquivo específico
Linha de execução
Árvore binária
Sistema operacional
Unidade de processamento dependente
Máquina específica alvo
104
Arquitetura
Arquivos executáveis portáteis
Pilhas
Árvore binária
Base de linha de execução
102
Unidade de processamento
Nível superior
Virtualização
Nível inferior

FIG. 3

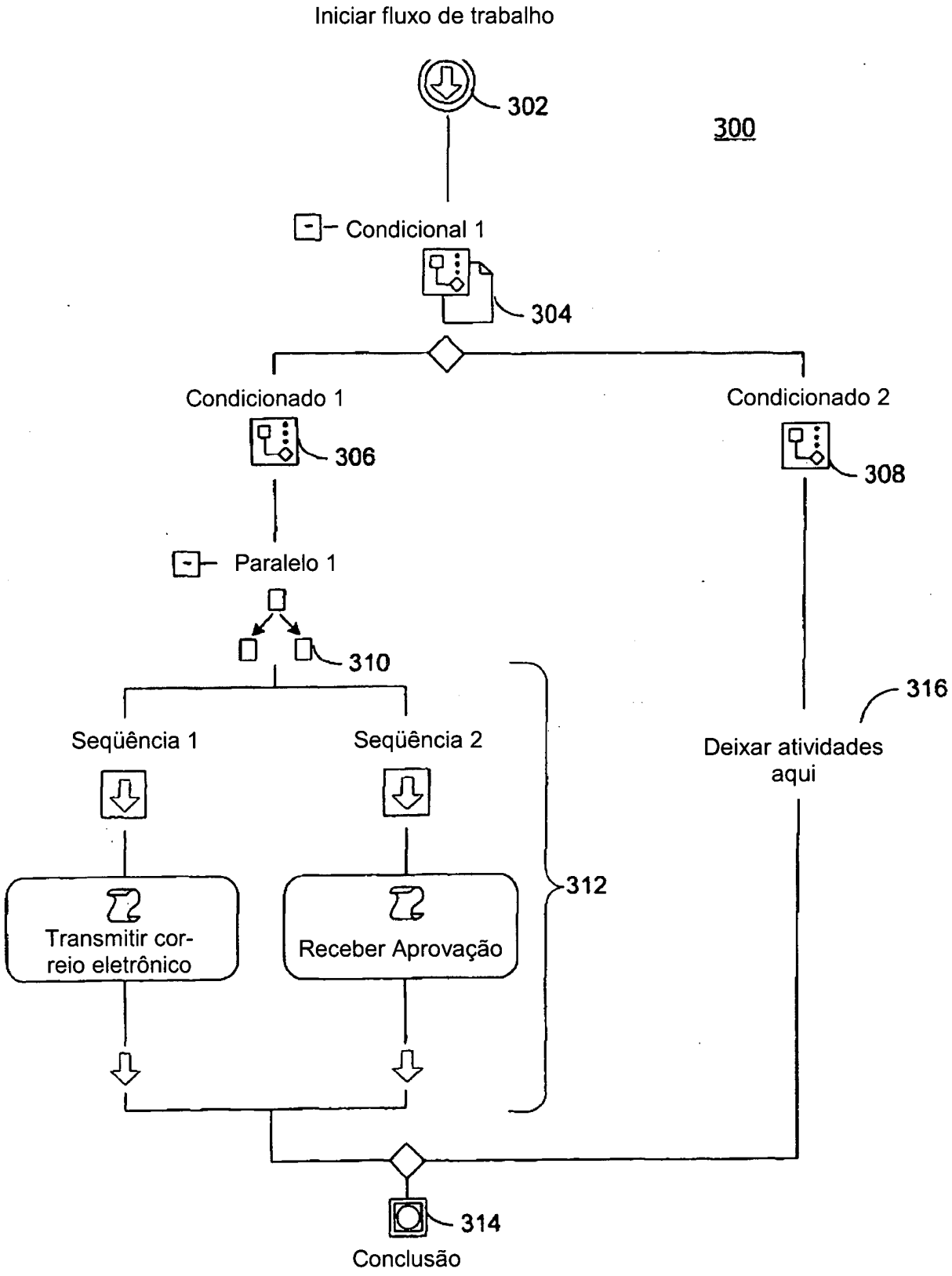

Iniciar fluxo de trabalho
302
300
Condicional 1
304
Condicionado 1
306
Condicionado 2
308
Paralelo 1
310
316
Seqüência 1
Seqüência 2
Deixar atividades aqui
Transmitir correio eletrônico
Receber Aprovação
312
314
Conclusão

FIG. 4

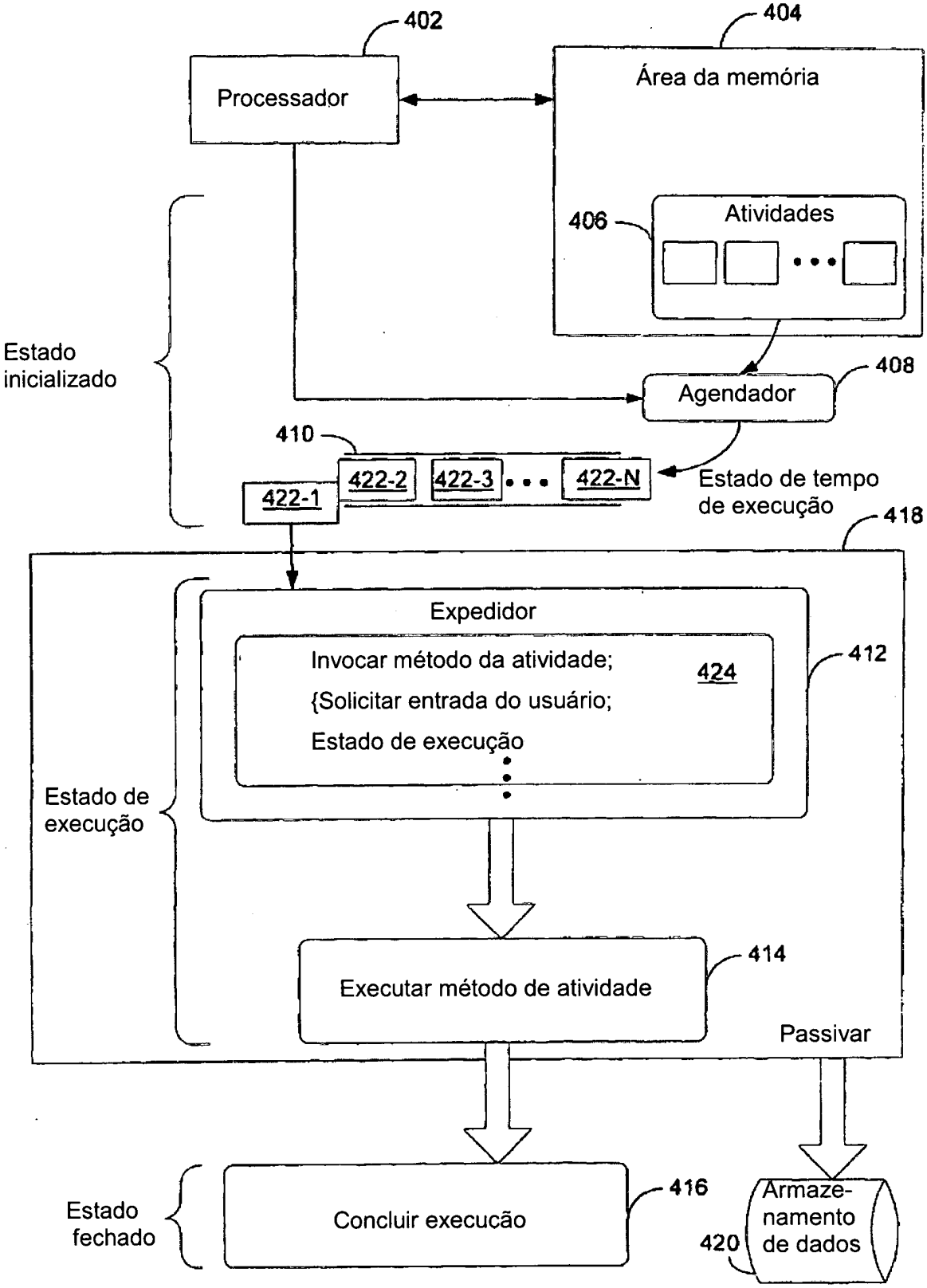
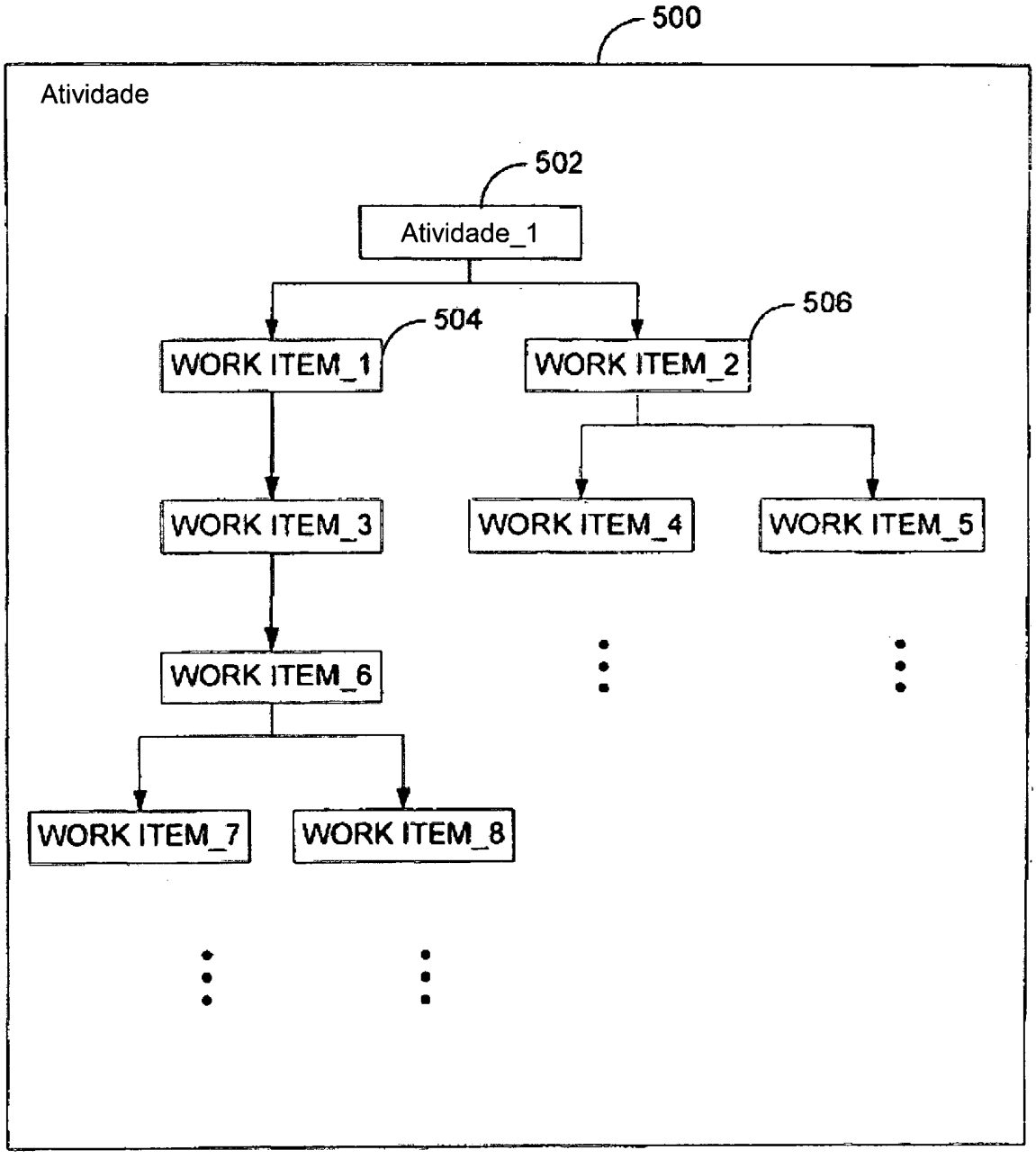

402
404
Processador
Área da memória
406
Atividades
Estado inicializado
408
Agendador
410
422-1
422-2
422-3
422-N
Estado de tempo de execução
418
Expedidor
Invocar método da atividade;
{Solicitar entrada do usuário;
Estado de execução
424
412
Estado de execução
414
Executar método de atividade
Passivar
Estado fechado
Concluir execucão
416
Armaze-namento de dados
420

FIG. 5


500
Atividade
502
Atividade_1
504
506
WORK ITEM_1
WORK ITEM_2
WORK ITEM_3
WORK ITEM_4
WORK ITEM_5
WORK ITEM_6
WORK ITEM_7
WORK ITEM_8

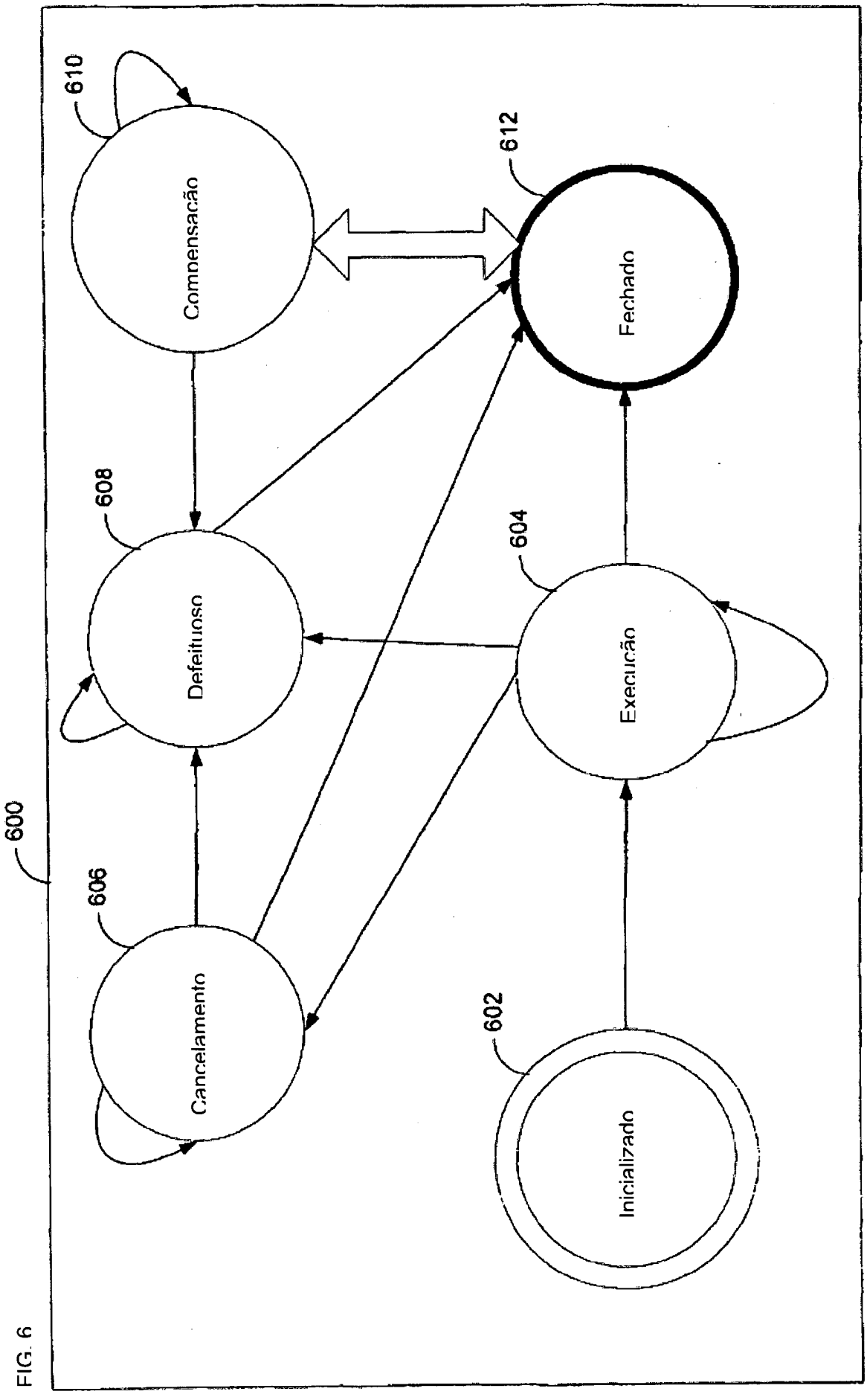

600
602
Inicializado
604
Execução
606
Cancelamento
608
Defeituoso
610
Compensação
612
Fechado

FIG. 6

FIG. 7

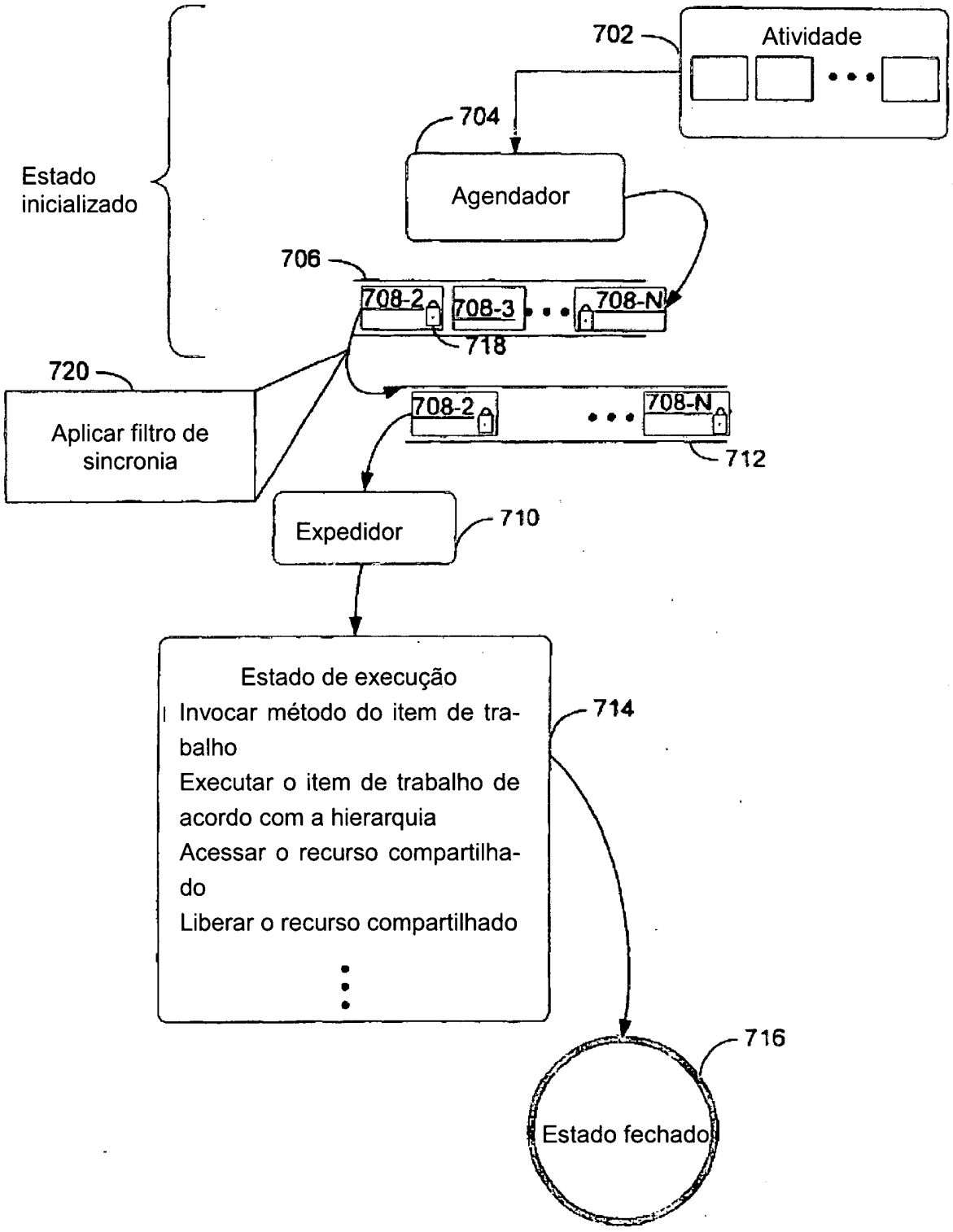

Estado inicializado
702
Atividade
704
Agendador
706
708-2
708-3
708-N
718
720
Aplicar filtro de sincronia
708-2
708-N
712
Expedidor
710
Estado de execução
Invocar método do item de trabalho
Executar o item de trabalho de acordo com a hierarquia
Acessar o recurso compartilhado
Liberar o recurso compartilhado
714
716
Estado fechado

FIG. 7B

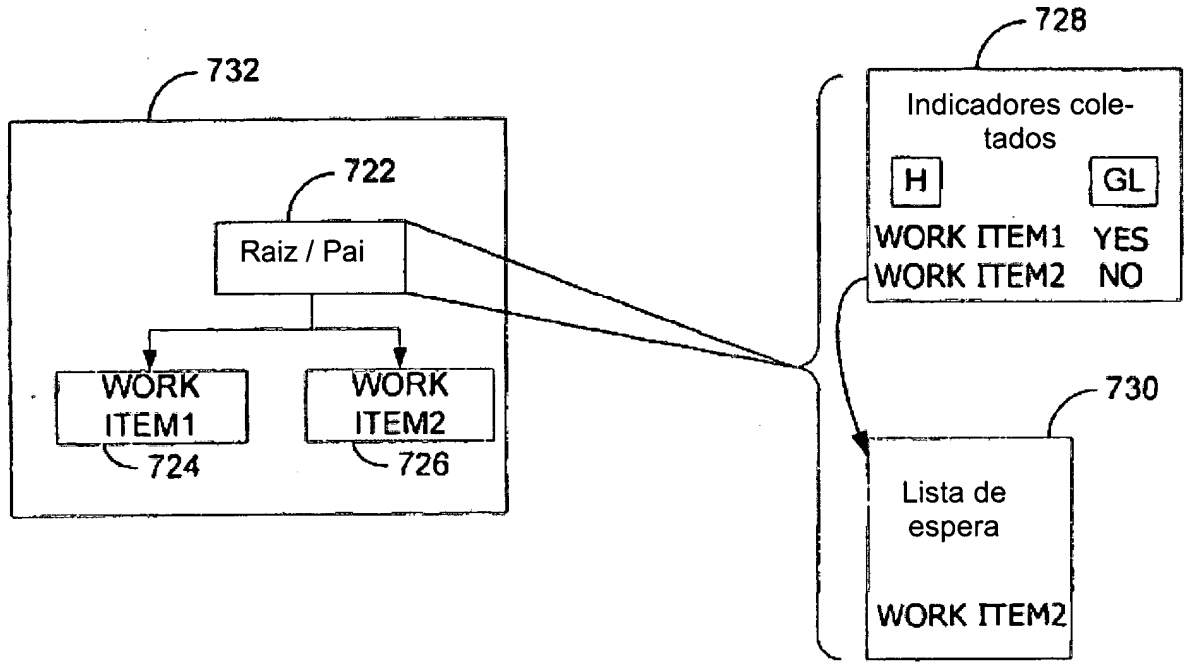

732
722
Raiz / Pai
WORK ITEM1
724
WORK ITEM2
726
728
Indicadores cole-tados
H
GL
WORK ITEM1 YES
WORK ITEM2 NO
730
Lista de espera
WORK ITEM2

FIG. 8

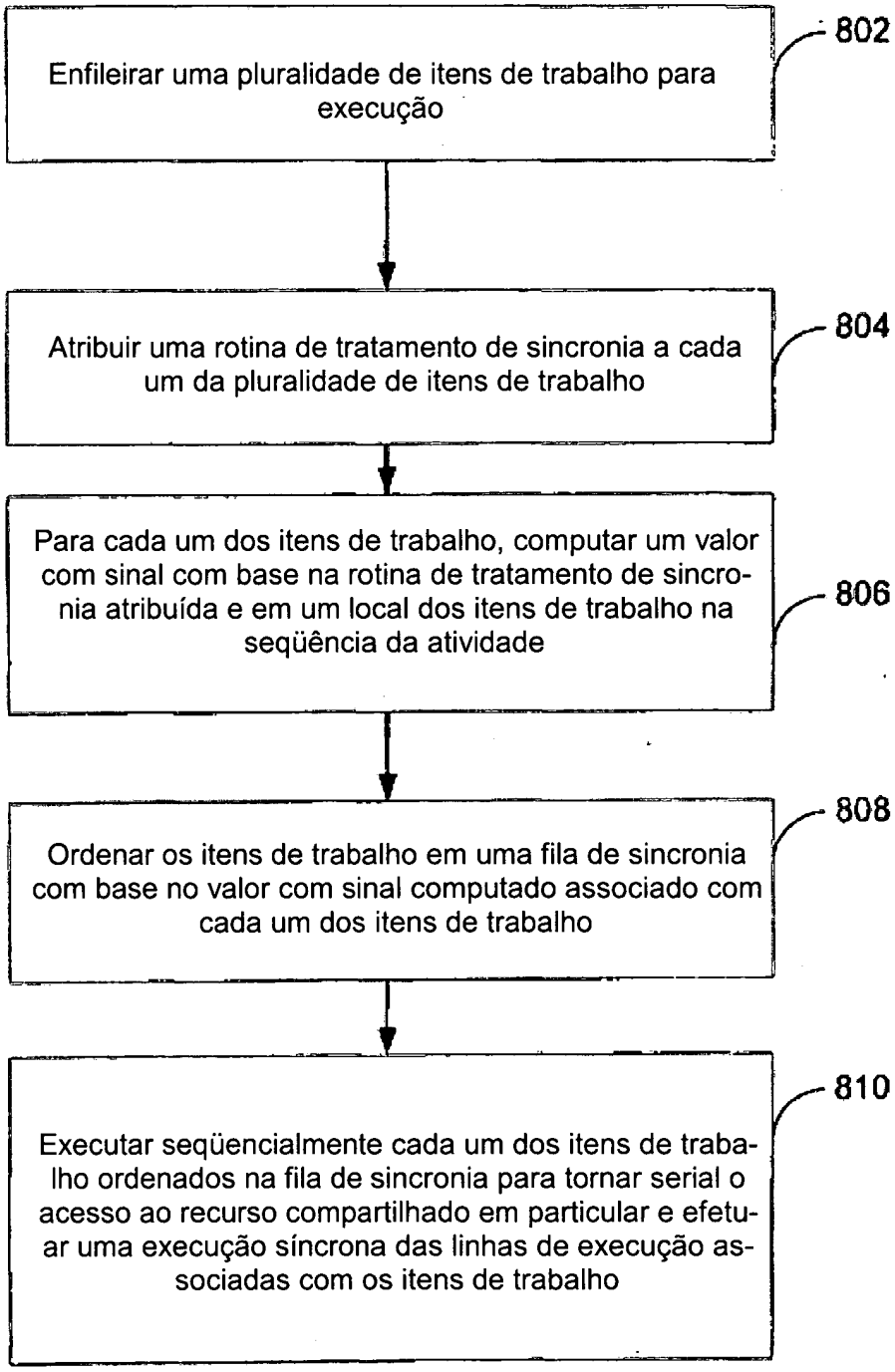
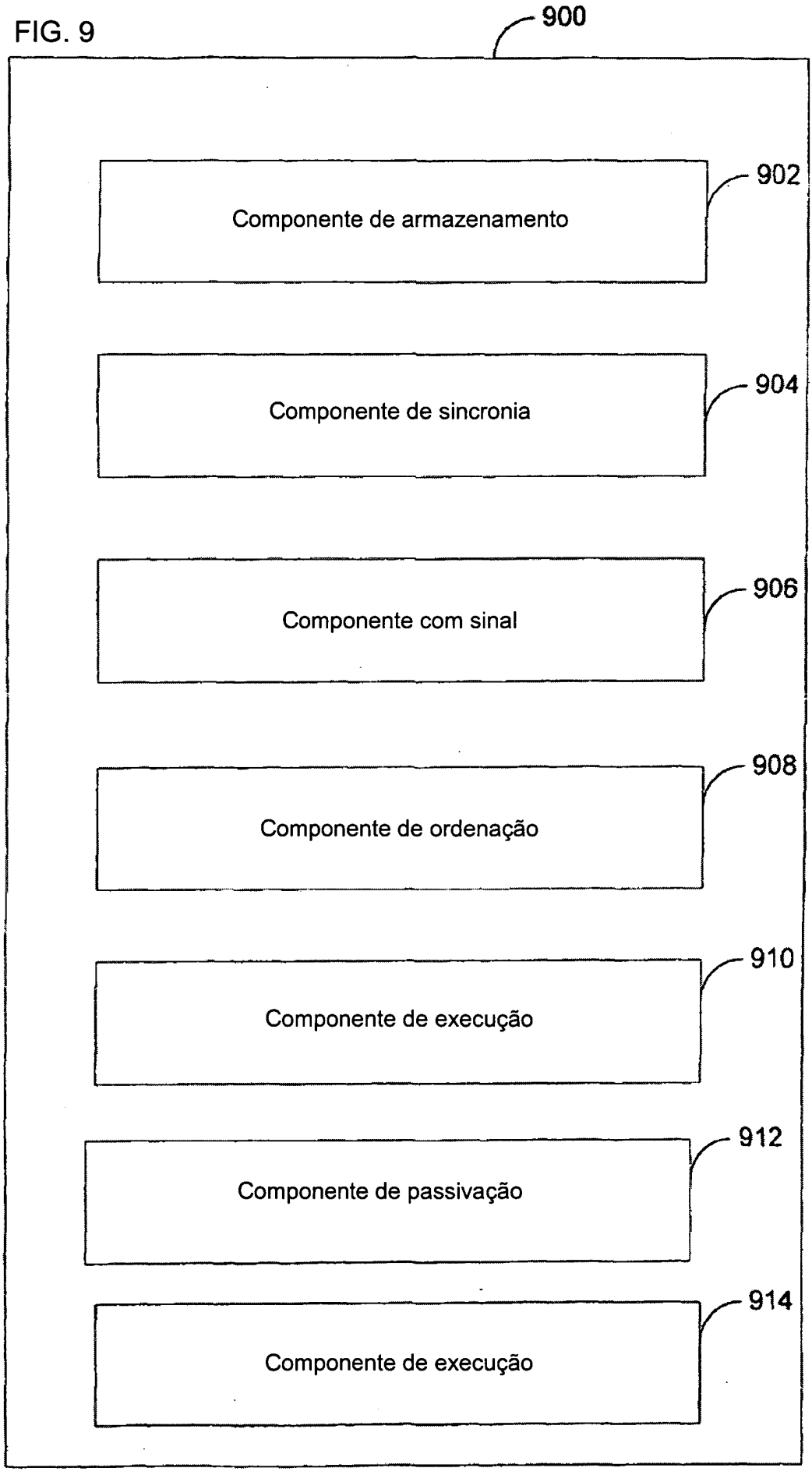

Enfileirar uma pluralidade de itens de trabalho para execução
802
Atribuir uma rotina de tratamento de sincronia a cada um da pluralidade de itens de trabalho
804
Para cada um dos itens de trabalho, computar um valor com sinal com base na rotina de tratamento de sincronia atribuída e em um local dos itens de trabalho na seqüência da atividade
806
Ordenar os itens de trabalho em uma fila de sincronia com base no valor com sinal computado associado com cada um dos itens de trabalho
808
Executar seqüencialmente cada um dos itens de trabalho ordenados na fila de sincronia para tornar serial o acesso ao recurso compartilhado em particular e efetuar uma execução síncrona das linhas de execução associadas com os itens de trabalho
810

FIG. 9


900
Componente de armazenamento
902
Componente de sincronia
904
Componente com sinal
906
Componente de ordenação
908
Componente de execução
910
Componente de passivação
912
Componente de execução
914

PI0710036-1

RESUMO

"MODELO DECLARATIVO PARA CONTROLE SIMULTÂNEO ATRAVÉS DE LINHAS DE EXECUÇÃO LEVES"

É divulgada a implementação da sincronia entre linhas de execução em um fluxo de trabalho. Uma área de memória armazena uma pluralidade de itens de trabalho em uma fila do agendador. Os itens de trabalho são associados com uma atividade no fluxo de trabalho, e cada item de trabalho é associado com uma linha de execução. Um processador é configurado para atribuir uma rotina de tratamento de sincronia a cada um dos itens de trabalho. A rotina de tratamento de sincronia indica um recurso compartilhado em particular a ser acessado pelos itens de trabalho. Um valor com sinal é computado para cada item de trabalho com base na rotina de tratamento de sincronia atribuída e nos itens de trabalho em uma hierarquia na atividade. Os itens de trabalho são ordenados em uma fila de sincronia com base no valor com sinal associado com cada item de trabalho. O processador executa seqüencialmente cada um dos itens de trabalho armazenados na fila de sincronia para tornar serial o acesso ao recurso compartilhado em particular e efetuar uma execução síncrona das linhas de execução associadas com os itens de trabalho.