República Federativa do Brasil
Ministério da Economia
Instituto Nacional da Propriedade Industrial

(11) BR 112019004964-9 B1

(22) Data do Depósito: 08/09/2017

(45) Data de Concessão: 12/01/2021

(54) Título: GERADOR DE CASO DE TESTE CONSTRUÍDO EM EDITOR DE FLUXO DE TRABALHO DE INTEGRAÇÃO DE DADOS

(51) Int.Cl.: G06F 9/44; G06F 11/36.

(30) Prioridade Unionista: 15/09/2016 US 62/395,179; 21/12/2016 US 15/386,930; 30/09/2016 US 62/402,880.

(73) Titular(es): TALEND, INC..

(72) Inventor(es): MICHAËL GUILLAUME MAURICE HIRT; CIARAN DYNES.

(86) Pedido PCT: PCT US2017050784 de 08/09/2017

(87) Publicação PCT: WO 2018/052813 de 22/03/2018

(85) Data do Início da Fase Nacional: 14/03/2019

(57) Resumo: A presente invenção refere-se a técnicas para a geração de casos de teste para os módulos da transformação que fazem parte de um fluxo de trabalho de integração de dados maior para um projeto de extrato, transferência e carga (ETL). Um caso de teste gerado de acordo com a presente invenção pode ser executado independentemente do aplicativo que gerou o mesmo e independentemente das origens e dos destinos dos dados mencionados no fluxo de trabalho de integração de dados. Para obter essa independência, o caso de teste pode incluir um código que simula as origens e os destinos dos dados que interagem com o módulo de transformação no fluxo de trabalho de integração de dados. Além disso, o caso de teste pode ser um teste de unidade baseado em uma estrutura de teste de unidade e pode ser compatível com uma estrutura de software selecionada.

Relatório Descritivo da Patente de Invenção para **"GERADOR DE CASO DE TESTE CONSTRUÍDO EM EDITOR DE FLUXO DE TRABALHO DE INTEGRAÇÃO DE DADOS".**

ANTECEDENTES DA INVENÇÃO

CAMPO DA INVENÇÃO

[001] A presente invenção refere-se de maneira geral à geração de testes de unidade para os elementos de um fluxo de trabalho de integração de dados. Mais especificamente, a presente invenção apresenta técnicas para um editor de fluxo de trabalho de integração de dados para a geração de código a fim de testar os módulos de programa selecionados por um usuário dentro de uma estrutura de software selecionada por um usuário.

DESCRIÇÃO DA TÉCNICA

[002] A tecnologia de obtenção e armazenamento de dados melhorou muito nas últimas décadas. Em particular, as taxas de coleta de dados, as velocidades de acesso aos dados e a capacidade de armazenamento dos dados avançaram consideravelmente. Além disso, as velocidades do processador do computador também aumentaram devido a várias questões de magnitude e um grande número de recursos computacionais tornaram-se prontamente disponíveis em nuvem. Em consequência, a análise de dados em grande escala que seria pouco prática há algumas décadas agora é possível.

[003] Muitos negócios modernos, universidades, governos e outras entidades coletam dados para fins de contabilidade, pesquisa, inteligência, marketing, inventário, controle de qualidade, transações e outras finalidades. O termo "Big Data" foi inventado para se referir às grandes quantidades de dados (por exemplo, terabytes) que tal entidade pode possuir. Essa grande quantidade de dados pode, estatisticamente, ser analisada para determinar tendências e pode ser usada para criar modelos previsíveis úteis (por exemplo, modelos de aprendizagem de

máquina). Uma entidade pode usar tais análises e modelos para se informar sobre a tomada de decisões e para identificar tendências, problemas e oportunidades potenciais.

[004] Frequentemente, a grande quantidade de dados que uma entidade possui é distribuída através de muitas unidades de armazenamento de dados diferentes. As unidades de armazenamento de dados diferentes, por sua vez, podem conter diferentes tipos de unidade de armazenamento de dados dos em diferentes formatos. Em consequência, os dados podem ter de ser extraídos de locais diferentes, reformatados, combinados e carregados em um único repositório de dados de modo que análises estatísticas possam ser executadas e modelos previsíveis possam ser criados. Este processo frequentemente é indicado como extrair, transformar e carregar (ETL).

BREVE DESCRIÇÃO DOS DESENHOS

[005] A Figura 1 ilustra um ambiente de computação exemplificador que pode ser usado para aplicar as técnicas da presente invenção, de acordo com uma modalidade.

[006] A Figura 2 ilustra uma vista mais detalhada de um sistema de computação, de acordo com uma modalidade.

[007] A Figura 3 ilustra um modelo gráfico de um fluxo de trabalho de integração de dados, tal como mostrado em uma tela, de acordo com uma modalidade.

[008] A Figura 4 é um fluxograma que ilustra um método para criação de um caso de teste para um módulo de transformação de um fluxo de trabalho de integração de dados, de acordo com uma modalidade.

[009] A Figura 5 é um fluxograma que ilustra um método para execução de um caso de teste para um módulo de transformação de um fluxo de trabalho de integração de dados, de acordo com uma modalidade.

[0010] A Figura 6 ilustra um sistema de integração de dados exemplificador que gera o código para os casos de teste, de acordo com uma modalidade.

DESCRIÇÃO DETALHADA DA INVENÇÃO

[0011] As modalidades aqui descritas descrevem técnicas para a geração de um caso de teste de unidade para um componente de um fluxo de trabalho de transformação de dados. O editor de fluxo de trabalho de integração de dados exibe um modelo gráfico do fluxo de trabalho de transformação de dados a um usuário. O usuário seleciona pelo menos um ícone do modelo gráfico que representa um módulo de transformação dentro do fluxo de trabalho de transformação de dados. O usuário seleciona uma opção para criar um caso de teste para o módulo de transformação. O usuário também pede que o caso de teste seja compatível com a estrutura de software especificada. O editor de fluxo de trabalho de integração de dados identifica caminhos de entrada para o módulo de transformação e um caminho de saída do módulo de transformação no fluxo de trabalho de transformação de dados. Um gerador de código cria um caso de teste para o módulo de transformação. O caso de teste pode ser executado ao usar a estrutura de software.

[0012] Quando executado, o caso de teste executa o módulo de transformação ao usar os dados de entrada de teste (por exemplo, fornecidos pelo usuário). O caso de teste então compara os dados de saída reais do módulo de transformação com os dados de saída de destino (por exemplo, fornecidos pelo usuário). Se os dados de saída reais combinarem com os dados de saída de destino, o caso de teste indica que o teste foi bem-sucedido. Em caso contrário, o caso de teste indica que o teste falhou e pode opcionalmente fornecer detalhes adicionais sobre a falha.

[0013] A fim de gerenciar eficazmente o software de ETL, a

integração contínua pode ser usada. O termo "integração contínua" refere-se a uma prática de desenvolvimento de software na qual os programadores de computador de uma equipe integram seu trabalho normalmente e verificam cada integração por meio de uma configuração automatizada, de modo que os erros de integração podem ser prontamente detectados.

[0014] Um fluxo de trabalho de integração de dados para um projeto de ETL pode especificar as fontes de dados de entrada, os módulos (por exemplo, do código de programação) que processam os dados de entrada e o destino para onde a saída do módulo é enviada, bem como a relação entre as fontes, os módulos e os destinos. O termo "módulo de transformação de dados" refere-se a um componente de um fluxo de trabalho de integração de dados que recebe dados de entrada de uma ou mais fontes de entrada, executa pelo menos uma operação nos dados de entrada para produzir os dados de saída e envia os dados de saída a um destino.

[0015] Os programadores de uma equipe de desenvolvimento de software que trabalham em um projeto de ETL podem mudar com o tempo. Frequentemente, os membros da equipe podem ser obrigados a modificar os componentes estranhos a um fluxo de trabalho de integração de dados existente. A estrutura ou a operação correta desses componentes estranhos podem não ser óbvias. Sob tais circunstâncias, um membro da equipe pode inadvertidamente introduzir erros no código de computador desses componentes. Alguns desses erros podem não ser detectados até que o projeto de ETL apresente resultados incorretos na produção. O teste de unidade permite que os desenvolvedores testem o comportamento esperado ou os estados de erro conhecidos, mas a codificação manual dos testes de unidade pode uma tarefa demorada e complexa.

[0016] As modalidades aqui descritas apresentam técnicas para a

geração de um caso de teste para um módulo de transformação sem a necessidade de codificação manual. O caso de teste pode ser executado independentemente do aplicativo que o gerou. O caso de teste pode ser executado sem o acesso às fontes de dados que fornecem a entrada ao módulo de transformação no fluxo de trabalho de integração de dados. Além disso, o caso de teste pode ser executado sem o acesso aos destinos que recebem a saída do módulo de transformação no fluxo de trabalho de integração de dados. Por essas razões, o caso de teste é altamente portável e é independente das origens dos dados e dos destinos indicadas no fluxo de trabalho de integração de dados.

[0017] Um caso de teste pode simular de modo preciso uma variedade de origens de dados e de estruturas, permitindo desse modo que o desenvolvedor teste o módulo de transformação virtualmente em relação aos conjuntos de dados de entrada que sejam inapropriados ou falsos a configuração ou a manutenção dos sistemas de bancos de dados reais para finalidades de teste. O desenvolvedor pode validar cada mudança em um módulo de transformação em relação ao caso de teste (por exemplo, clicando em uma tecla ou selecionando uma opção para rodar o caso de teste automaticamente cada vez que o módulo de transformação é modificado).

[0018] O caso de teste também pode ser associado com uma versão do fluxo de trabalho de integração de dados. Se a funcionalidade pretendida do módulo de transformação mudar em versões mais atrasadas, o desenvolvedor pode determinar que o caso de teste está obsoleto, observando que o caso de teste é associado com uma versão anterior.

[0019] A Figura 1 ilustra um ambiente de computação exemplificador 100 que pode ser usado para aplicar as técnicas da presente invenção, de acordo com uma modalidade. Um sistema de

computação 108, uma unidade de armazenamento de dados 104 e uma unidade de armazenamento de dados 106 são conectadas à rede 102. O editor de fluxo de trabalho de integração de dados 110 indica um modelo gráfico do fluxo de trabalho de integração de dados 112 na tela 118.

[0020] Um usuário seleciona pelo menos um ícone no modelo gráfico e pede que um caso de teste que use a estrutura de teste de unidade 114 seja criado para um módulo de transformação representado pelo ícone selecionado. O editor de fluxo de trabalho de integração de dados 110 determina que o módulo de transformação no fluxo de trabalho de integração de dados 112 seja configurado para receber a entrada da unidade de armazenamento de dados 104 e enviar a saída à unidade de armazenamento de dados 106. O editor de fluxo de trabalho de integração de dados 110 então gera o caso de teste 116.

[0021] O caso de teste 116, quando roda, executa o módulo de transformação na entrada de teste. O caso de teste 116 pode receber a entrada de teste de uma fonte que não seja a unidade de armazenamento de dados 104, e já fornece a entrada de teste ao módulo de transformação de uma maneira que imita a unidade de armazenamento de dados 104. Em consequência, a unidade de armazenamento de dados 104 não tem de ser acessada quando o caso de teste 116 é executado. Do mesmo modo, o caso de teste 116 pode receber a saída do módulo de transformação de uma maneira que imita a unidade de armazenamento de dados 106. Em consequência, a unidade de armazenamento de dados 106 não tem de ser acessada quando o caso de teste 116 é executado.

[0022] O caso de teste 116 utiliza a estrutura de teste de unidade 114 e pode ser executado em uma estrutura de software designada pelo usuário. A estrutura de teste de unidade 114 (por exemplo, tal como Junit) pode ser associada a uma linguagem de programação (por

exemplo, tal como Java). A estrutura de software pode ser Apache Hadoop ou alguma outra estrutura. O caso de teste 116 pode ser executado utilizando a estrutura de software mesmo se o editor de fluxo de trabalho de integração de dados 110 não estiver sendo executado. Além disso, o caso de teste 116 é portável a outros sistemas de computação que podem usar a estrutura de teste de unidade 114 e a estrutura de software. Em outras palavras, o caso de teste 116 fica livre de dependências no editor de fluxo de trabalho de integração de dados 110.

[0023] O usuário também pode designar que o caso de teste seja executado automaticamente ao usar a entrada de teste especificada e a saída de destino especificada cada vez que o módulo de transformação é modificado. Usar o caso de teste dessa maneira permite que os desenvolvedores de software verifiquem se o módulo de transformação ainda funciona corretamente depois que as mudanças são feitas.

[0024] A Figura 2 ilustra uma vista mais detalhada do sistema de computação 108, de acordo com uma modalidade. O sistema de computação 108, uma unidade de armazenamento de dados 104 e uma unidade de armazenamento de dados 106 são conectados a uma rede 102. O editor de fluxo de trabalho de integração de dados 110 inclui uma interface gráfica de usuário (GUI) 202, um gerador de código 204 e um modelo gráfico 206 do fluxo de trabalho de integração de dados 112. O modelo gráfico 206 inclui os ícones 208. O editor de fluxo de trabalho de integração de dados 110 exibe um modelo gráfico 206 do fluxo de trabalho de integração de dados 112 na GUI 202 na tela 118.

[0025] O usuário pode selecionar um ou mais ícones 208 no modelo gráfico 206 para designar um módulo de transformação 210 do fluxo de trabalho de integração de dados 112 para teste. O editor de fluxo de trabalho de integração de dados 110 determina que o módulo de

transformação 210 seja configurado para receber a entrada da unidade de armazenamento de dados 104 e para enviar a saída à unidade de armazenamento de dados 106. O gerador de código 204 então gera o caso de teste 116 ao usar a estrutura de teste de unidade 114.

[0026] O caso de teste 116, quando roda, executa o módulo de transformação 210 na entrada de teste. O caso de teste 116 pode receber a entrada de teste de uma fonte que não seja a unidade de armazenamento de dados 104, e já fornece a entrada de teste ao módulo de transformação de uma maneira que imita a unidade de armazenamento de dados 104. Em consequência, a unidade de armazenamento de dados 104 não tem de ser acessada quando o caso de teste 116 é executado. Do mesmo modo, o caso de teste 116 pode receber a saída do módulo de transformação de uma maneira que imita a unidade de armazenamento de dados 106. Em consequência, a unidade de armazenamento de dados 106 não tem de ser acessada quando o caso de teste 116 é executado.

[0027] O caso de teste 116 utiliza estrutura de teste de unidade 114 e pode ser executado em uma estrutura de software designada pelo usuário. A estrutura de teste de unidade 114 pode ser associada a uma linguagem de programação (por exemplo, tal como Junit, que é associada ao Java). Por exemplo, a estrutura de software pode ser a Apache Hadoop ou a estrutura Spark. O caso de teste 116 pode ser executado ao usar a estrutura de software mesmo se o editor de fluxo de trabalho de integração de dados 110 não estiver sendo executado. Além disso, o caso de teste 116 é portável a outros sistemas de computação que podem usar a estrutura de teste de unidade 114 e a estrutura de software. Em outras palavras, o caso de teste 116 fica livre de dependências no editor de fluxo de trabalho de integração de dados 110.

[0028] A Figura 3 ilustra um modelo gráfico 300 de um fluxo de

trabalho de integração de dados tal como mostrado na tela 302, de acordo com uma modalidade. O ícone 304 representa um módulo de transformação. O ícone 306 e o ícone 310 representam as fontes de dados das quais o módulo de transformação recebe a entrada no fluxo de trabalho de integração de dados. Por exemplo, o ícone 306 pode representar um banco de dados, enquanto o ícone 310 pode representar um arquivo no formato de texto plano. O ícone 314 representa uma fonte de dados à qual o módulo de transformação fornece a saída.

[0029] A seta 312 é exibida no modelo gráfico 300 para indicar a um usuário que o módulo de transformação representado pelo ícone 304 recebe a entrada da fonte de dados representada pelo ícone 310. Do mesmo modo, a seta 308 é exibida no modelo gráfico 300 para indicar ao usuário que o módulo de transformação representado pelo ícone 304 também recebe a entrada da fonte de dados representada pelo ícone 306. A seta 316 é exibida no modelo gráfico 300 para indicar ao usuário que o módulo de transformação representado pelo ícone 304 fornece a saída à fonte de dados representada pelo ícone 314.

[0030] O usuário pode selecionar o ícone 304 ao usar o cursor 318. Uma vez que o ícone 304 é selecionado, uma janela drop-down pode aparecer para fornecer uma opção a fim de criar um caso de teste. O usuário pode selecionar a opção e fornecer detalhes adicionais para o caso de teste, tal como uma versão do fluxo de trabalho de integração de dados para associar com o teste e uma estrutura de software para o caso de teste. Com base nessas entradas do usuário, um gerador de código cria um caso de teste para o módulo de transformação ao usar uma estrutura de teste de unidade.

[0031] O código incluído no caso de teste é projetado para receber os dados de entrada de teste de fontes que não sejam os repositórios de dados representados pelo ícone 306 e pelo ícone 310. No entanto, quando o caso de teste é executado, ele fornece os dados de entrada

de teste ao módulo de transformação que está sendo testado de uma maneira que ainda permita que o módulo de transformação seja executado normalmente. Por exemplo, se o módulo de transformação for projetado para receber a entrada em um determinado formato ou em uma certa estrutura de dados, o caso de teste inclui o código que pode fornecer os dados de entrada de teste nesse formato ou estrutura de dados. Do mesmo modo, se o módulo de transformação fornecer a saída ao repositório de dados representado pelo ícone 314 em um determinado formato ou estrutura de dados, o caso de teste inclui o código para receber continuamente a saída nesse formato ou estrutura de dados. O caso de teste também inclui o código que compara a saída do módulo de transformação com a saída de destino.

[0032] A Figura 4 é um fluxograma que ilustra um método 400 para criação de um caso de teste para um módulo de transformação de um fluxo de trabalho de integração de dados, de acordo com uma modalidade. Na etapa 402, um editor de fluxo de trabalho de integração de dados exibe um modelo gráfico de um fluxo de trabalho de integração de dados na interface gráfica de usuário.

[0033] Na etapa 404, o editor de fluxo de trabalho de integração de dados identifica um módulo de transformação selecionado por um usuário no modelo gráfico. Um ou mais ícones no modelo gráfico podem representar o módulo de transformação.

[0034] Na etapa 406, o editor de fluxo de trabalho de integração de dados identifica um caminho de entrada ao módulo de transformação no fluxo de trabalho de integração de dados. O caminho de entrada pode ser representado por uma seta que aponta para um ícone que representa o módulo de transformação no modelo gráfico. Na etapa 408, o editor de fluxo de trabalho de integração de dados determina se há caminhos de entrada adicionais ao módulo de transformação e repete a etapa 406 para cada caminho de entrada adicional.

[0035] Na etapa 410, o editor de fluxo de trabalho de integração de dados identifica um caminho de saída do módulo de transformação no fluxo de trabalho de integração de dados. O caminho de saída pode ser representado por uma seta que aponta para longe do ícone que representa o módulo de transformação no modelo gráfico. Na etapa 412, o editor de fluxo de trabalho de integração de dados determina se há caminhos de saída adicionais do módulo de transformação e repete a etapa 410 para cada caminho de saída adicional.

[0036] Na etapa 414, o editor de fluxo de trabalho de integração de dados identifica uma estrutura de software selecionada pelo usuário. Na etapa 416, um gerador de código gera o código que define um caso de teste para o módulo de transformação selecionado. O caso de teste inclui o código para receber os dados de entrada de teste para cada caminho de entrada identificado nas etapas 406 a 408 de uma fonte de dados especificada pelo usuário em vez das fontes de dados reais conectadas aos caminhos de entrada no fluxo de trabalho de integração de dados. O caso de teste também inclui o código para fornecer os dados de entrada de teste ao módulo de transformação de uma maneira que imita as fontes de dados reais quando o caso de teste executa o módulo de transformação.

[0037] Além disso, o caso de teste inclui o código para receber os dados de saída de destino para cada caminho de saída identificado nas etapas 410 a 412. O caso de teste também inclui o código para receber os dados de saída reais do módulo de transformação para cada caminho de saída quando o caso de teste executa o módulo de transformação como parte do processo de teste. O caso de teste inclui o código para comparar os dados de saída de destino com os dados de saída reais para cada caminho. Se houver qualquer discrepância entre os dados de saída de destino e os dados de saída reais, o caso de teste indica que o módulo de transformação falhou em pelo menos parte do

teste.

[0038] A Figura 5 é um fluxograma que ilustra um método 500 para executar um caso de teste para um módulo de transformação de um fluxo de trabalho de integração de dados, de acordo com uma modalidade. O caso de teste pode ser um teste de unidade que empregue uma estrutura de teste de unidade e seja compatível com a estrutura de software.

[0039] Na etapa 502, o caso de teste recebe os dados de entrada de teste para o módulo de transformação. Os dados de entrada de teste podem ser recebidos em um arquivo no formato de texto plano ou em algum outro formato. Na etapa 504, o caso de teste recebe os dados de saída de destino. Assim como os dados de entrada de teste, os dados de entrada de destino podem ser recebidos em um arquivo no formato de texto plano ou em algum outro formato. Os dados de saída de destino indicam o que o módulo de transformação deverá produzir quando o módulo de transformação for executado ao usar os dados de entrada de teste como entrada.

[0040] Na etapa 506, o caso de teste simula uma primeira fonte de dados para fornecer os dados de entrada de teste ao módulo de transformação (em que o caso de teste está sendo executado como parte do processo de teste). Dentro do fluxo de trabalho de integração de dados, o módulo de transformação é configurado para receber a entrada da primeira fonte de dados. No entanto, quando o módulo de transformação é executado pelo caso de teste, o caso de teste não permite que o módulo de transformação acesse a primeira fonte de dados. Em vez disso, o caso de teste simula a primeira fonte de dados ao fazer interface com o módulo de transformação da mesma maneira que a primeira fonte de dados faria se fornecesse os dados de entrada de teste ao módulo de transformação. Isso permite que o caso de teste seja independente da primeira fonte de dados.

[0041] Na etapa 508, o caso de teste simula uma segunda fonte de dados para receber os dados de saída reais do módulo de transformação. Dentro do fluxo de trabalho de integração de dados, o módulo de transformação é configurado para enviar a saída à primeira fonte de dados. No entanto, quando o módulo de transformação é executado pelo caso de teste, o caso de teste não permite que o módulo de transformação acesse a segunda fonte de dados. Em vez disso, o caso de teste simula a segunda fonte de dados ao fazer interface com o módulo de transformação da mesma maneira que a segunda fonte de dados faria se recebesse os dados de saída reais do módulo de transformação. Isso permite que o caso de teste seja independente da segunda fonte de dados.

[0042] Na etapa 510, o caso de teste compara os dados de saída reais com os dados de saída de destino. Na etapa 512, o caso de teste fornece os resultados de teste com base na comparação. Se os dados de saída reais corresponderem aos dados de saída, os resultados do teste indicam que o módulo de transformação passou no teste. Em caso contrário, os resultados do teste indicam que o módulo de transformação falhou no teste. Uma indicação de falha ou não do teste pode ser associada com o teste e ser armazenada.

[0043] Na etapa 514, um editor de fluxo de trabalho de integração de dados em que o caso de teste foi criado determina se o módulo de transformação foi alterado desde a última vez em que o caso de teste foi executado. Se o módulo de transformação tiver sido alterado, as etapas 502 a 512 são repetidas. Em caso contrário, o método 500 termina. Em uma modalidade, a etapa 514 é opcional. O usuário pode selecionar uma opção no editor de fluxo de trabalho de integração de dados que pede que o caso de teste seja executado cada vez que o módulo de transformação for alterado.

[0044] A Figura 6 ilustra um sistema de integração de dados

exemplificador 600 que gera o código para os casos de teste, de acordo com uma modalidade. Tal como mostrado, o sistema de integração de dados 600 inclui uma unidade de processamento central (CPU) 602, uma ou mais interfaces de dispositivo de entrada/saída (E/S) 604 que pode permitir a conexão de vários dispositivos de E/S 614 (por exemplo, teclados, monitores, mouse, entrada de caneta, etc.) ao sistema de integração de dados 600, uma interface de rede 606, uma memória 608, um armazenamento 610 e um interconector 612.

[0045] A CPU 602 pode recuperar e executar as instruções de programação armazenadas na memória 608. Do mesmo modo, a CPU 602 pode recuperar e armazenar os dados do aplicativo localizados na memória 608. O interconector 612 transmite instruções de programação e dados do aplicativo entre a CPU 602, o dispositivo de E/S, a interface 604, a interface de rede 606, a memória 608 e o armazenamento 610. A CPU 602 pode representar uma única CPU, várias CPUs, uma única CPU com vários núcleos de processamento e outros. Adicionalmente, a memória 606 representa a memória de acesso aleatório. Além disso, o armazenamento 610 pode ser uma unidade de disco. Embora mostrado como uma única unidade, o armazenamento 610 pode ser uma combinação de dispositivos de armazenamento fixos e/ou removíveis, tais como unidades de disco fixas, cartões de memória removíveis ou de armazenamento ótico, armazenamento de dados em rede (NAS) ou uma rede de área de armazenamento (SAN).

[0046] Tal como mostrado, a memória 608 inclui o editor de fluxo de trabalho de integração de dados 110. O editor de fluxo de trabalho de integração de dados 110 exibe um modelo gráfico do fluxo de trabalho de integração de dados 112. O usuário pode selecionar um ou mais ícones no modelo gráfico que representem um módulo de transformação dentro do fluxo de trabalho de integração de dados 112. O usuário também pode pedir que o editor de fluxo de trabalho de

integração de dados 110 gere o código para definir um teste de unidade para o modelo de transformação. Em resposta, o editor de fluxo de trabalho de integração de dados 110 cria o caso de teste 116 com base na estrutura de teste de unidade 114.

[0047] As descrições das várias modalidades da presente invenção foram apresentadas para fins de ilustração, mas não pretendem ser exaustivas ou limitadas às modalidades descritas. Muitas modificações e variações ficarão evidentes aos elementos versados no estado da técnica sem sair do âmbito e do caráter das modalidades descritas. A terminologia aqui utilizada foi escolhida para explicar melhor os princípios das modalidades, do aplicativo prático ou das melhorias técnicas em relação às tecnologias encontradas no mercado, ou para permitir que outros elementos versados na técnica compreendam as modalidades aqui descritas.

[0048] Os exemplos adicionais de geração de um caso de teste de unidade para um componente de um fluxo de trabalho de transformação de dados são fornecidos no apêndice anexo.

[0049] Embora o acima exposto tenha sido direcionado às modalidades da presente invenção, modalidades diferentes e adicionais da descrição podem ser contempladas sem sair do âmbito básico da mesma, e o âmbito da mesma é determinado pelas reivindicações a seguir.

REIVINDICAÇÕES

1. Método para a geração de um caso de teste para um módulo de transformação de um fluxo de trabalho de integração de dados, em que o método é, caracterizado pelo fato de que compreende:

recepcionar um pedido para a criação do caso de teste para o módulo de transformação do fluxo de trabalho de integração de dados que é compatível com uma estrutura de software;

identificar uma primeira passagem de rede de uma fonte de dados de entrada para o módulo de transformação no fluxo de trabalho de integração de dados;

identificar uma segunda passagem de rede do módulo de transformação para uma fonte de dados de saída no fluxo de trabalho de integração de dados;

gerar o código que define o caso de teste para o módulo de transformação;

executar o caso de teste na estrutura de software através de:

enviar dados de entrada de uma fonte simulada de entrada de dados do caso de teste para o módulo de transformação sem acessar a fonte de dados de entrada;

receber dados de saída em uma saída simulada de saída de dados do caso de teste do módulo de transformação sem acessar a fonte de dados de saída;

determinar que o módulo de transformação foi alterado para gerar um módulo alterado; e

reexecutar o caso de teste caso na estrutura de software, em que a reexecução do caso de teste é configurada para verificar uma funcionalidade do módulo alterado.

2. Método, de acordo com a reivindicação 1, caracterizado pelo fato de que também compreende:

exibir um modelo gráfico do fluxo de trabalho de integração

de dados em uma interface gráfica do usuário (GUI), em que pelo menos um ícone do modelo gráfico representa o módulo de transformação; e

recepcionar o pedido através da interface gráfica do usuário, em que selecionar pelo menos um ícone identifica o módulo de transformação.

3. Método, de acordo com a reivindicação 1, caracterizado pelo fato de que também compreende:

recepcionar os dados de entrada de teste para o caso de teste; e

recepcionar os dados de saída de destino para o caso de teste, em que executar o caso de teste compreende:

executar o módulo de transformação ao usar os dados de entrada de teste,

recepcionar os dados de saída reais do módulo de transformação, e

comparar os dados de saída reais com os dados de saída de destino.

4. Método, de acordo com a reivindicação 3, caracterizado pelo fato de que também compreende:

determinar, com base na comparação dos dados de saída reais com os dados de saída de destino, se o módulo de transformação passou ou falhou em um teste definido pelo caso de teste, pelos dados de entrada de teste, e pelos dados de saída de destino; e

armazenar uma indicação da determinação.

5. Método, de acordo com a reivindicação 3, caracterizado pelo fato de que a execução do caso de teste inclui a simulação da fonte de dados de entrada para prover os dados de entrada de teste ao módulo de transformação.

6. Método, de acordo com a reivindicação 1, caracterizado pelo fato de que o caso de teste é associado com uma versão do fluxo

de trabalho de integração de dados.

7. Meio de armazenamento não transitório que pode ser lido em computador que contém instruções que, quando executadas por um ou mais processadores, executa uma operação para a geração de um caso de teste para um módulo de transformação de um fluxo de trabalho de integração de dados, caracterizado pelo fato de que a operação compreende:

recepcionar um pedido para a criação do caso de teste para o módulo de transformação do fluxo de trabalho de integração de dados que é compatível com uma estrutura de software;

identificar uma primeira passagem de rede de uma fonte de dados de entrada para o módulo de transformação no fluxo de trabalho de integração de dados;

identificar uma segunda passagem de rede do módulo de transformação para uma fonte de dados de saída no fluxo de trabalho de integração de dados;

gerar o código que define o caso de teste para o módulo de transformação;

executar o caso de teste na estrutura de software através de:

enviar dados de entrada de uma fonte simulada de entrada de dados do caso de teste para o módulo de transformação sem acessar a fonte de dados de entrada;

receber dados de saída em uma saída simulada de saída de dados do caso de teste do módulo de transformação sem acessar a fonte de dados de saída;

determinar que o módulo de transformação foi alterado para gerar um módulo alterado; e

reexecutar o caso de teste caso na estrutura de software, em que a reexecução do caso de teste é configurada para verificar uma funcionalidade do módulo alterado.

8. Meio de armazenamento não transitório que pode ser lido em computador, como definido na reivindicação 7, caracterizado pelo fato de que a operação também compreende:

exibir um modelo gráfico do fluxo de trabalho de integração de dados em uma interface gráfica do usuário (GUI), em que pelo menos um ícone do modelo gráfico representa o módulo de transformação; e

recepcionar o pedido através da interface gráfica do usuário, em que selecionar pelo menos um ícone identifica o módulo de transformação.

9. Meio de armazenamento não transitório que pode ser lido em computador, como definido na reivindicação 7, caracterizado pelo fato de que a operação também compreende:

recepcionar os dados de entrada de teste para o caso de teste; e

recepcionar os dados de saída de destino para o caso de teste, em que executar o caso de teste compreende:

executar o módulo de transformação ao usar os dados de entrada de teste,

recepcionar os dados de saída reais do módulo de transformação, e

comparar os dados de saída reais com os dados de saída de destino.

10. Meio de armazenamento não transitório que pode ser lido em computador, de acordo com a reivindicação 9, caracterizado pelo fato de que a operação também compreende:

determinar, com base na comparação dos dados de saída reais com os dados de saída de destino, se o módulo de transformação passou ou falhou em um teste definido pelo caso de teste, pelos dados de entrada de teste, e pelos dados de saída de destino; e

armazenar uma indicação da determinação.

11. Meio de armazenamento não transitório que pode ser lido em computador, de acordo com a reivindicação 9, caracterizado pelo fato de que a execução do caso de teste inclui a simulação da fonte de dados de entrada para prover os dados de entrada de teste ao módulo de transformação.

12. Meio de armazenamento não transitório que pode ser lido em computador, de acordo com a reivindicação 7, caracterizado pelo fato de que o caso de teste é associado com uma versão do fluxo de trabalho de integração de dados.

13. Sistema para a geração de um caso de teste para um módulo de transformação de um fluxo de trabalho de integração de dados, o qual é, caracterizado pelo fato de que compreende:

um ou mais processadores; e

uma memória que armazena um ou mais aplicativos que, quando executado em um ou mais processadores, executa uma operação, em que a operação compreende:

recepcionar um pedido para a criação do caso de teste para o módulo de transformação do fluxo de trabalho de integração de dados que é compatível com uma estrutura de software,

identificar uma primeira passagem de rede de uma fonte de dados de entrada para o módulo de transformação no fluxo de trabalho de integração de dados;

identificar uma segunda passagem de rede do módulo de transformação para uma fonte de dados de saída no fluxo de trabalho de integração de dados;

gerar o código que define o caso de teste para o módulo de transformação;

executar o caso de teste na estrutura de software através de:

enviar dados de entrada de uma fonte simulada de entrada de dados do caso de teste para o módulo de

transformação sem acessar a fonte de dados de entrada;

receber dados de saída em uma saída simulada de saída de dados do caso de teste do módulo de transformação sem acessar a fonte de dados de saída;

determinar que o módulo de transformação foi alterado para gerar um módulo alterado; e

reexecutar o caso de teste caso na estrutura de software, em que a reexecução do caso de teste é configurada para verificar uma funcionalidade do módulo alterado.

14. Sistema, de acordo com a reivindicação 13, caracterizado pelo fato de que a operação também compreende:

exibir um modelo gráfico do fluxo de trabalho de integração de dados em uma interface gráfica do usuário (GUI), em que pelo menos um ícone do modelo gráfico representa o módulo de transformação; e

recepcionar o pedido através da interface gráfica do usuário, em que selecionar pelo menos um ícone identifica o módulo de transformação.

15. Sistema, de acordo com a reivindicação 13, caracterizado pelo fato de que a operação também compreende:

recepcionar os dados de entrada de teste para o caso de teste; e

recepcionar os dados de saída de destino para o caso de teste, em que executar o caso de teste compreende:

executar o módulo de transformação ao usar os dados de entrada de teste,

recepcionar os dados de saída reais do módulo de transformação, e

comparar os dados de saída reais com os dados de saída de destino.

16. Sistema, de acordo com a reivindicação 15,

caracterizado pelo fato de que a operação compreende:

determinar, com base na comparação dos dados de saída reais com os dados de saída de destino, se o módulo de transformação passou ou falhou em um teste definido pelo caso de teste, pelos dados de entrada de teste, e pelos dados de saída de destino; e

armazenar uma indicação da determinação.

17. Sistema, de acordo com a reivindicação 15, caracterizado pelo fato de que a execução do caso de teste inclui a simulação da fonte de dados de entrada para prover os dados de entrada de teste ao módulo de transformação.

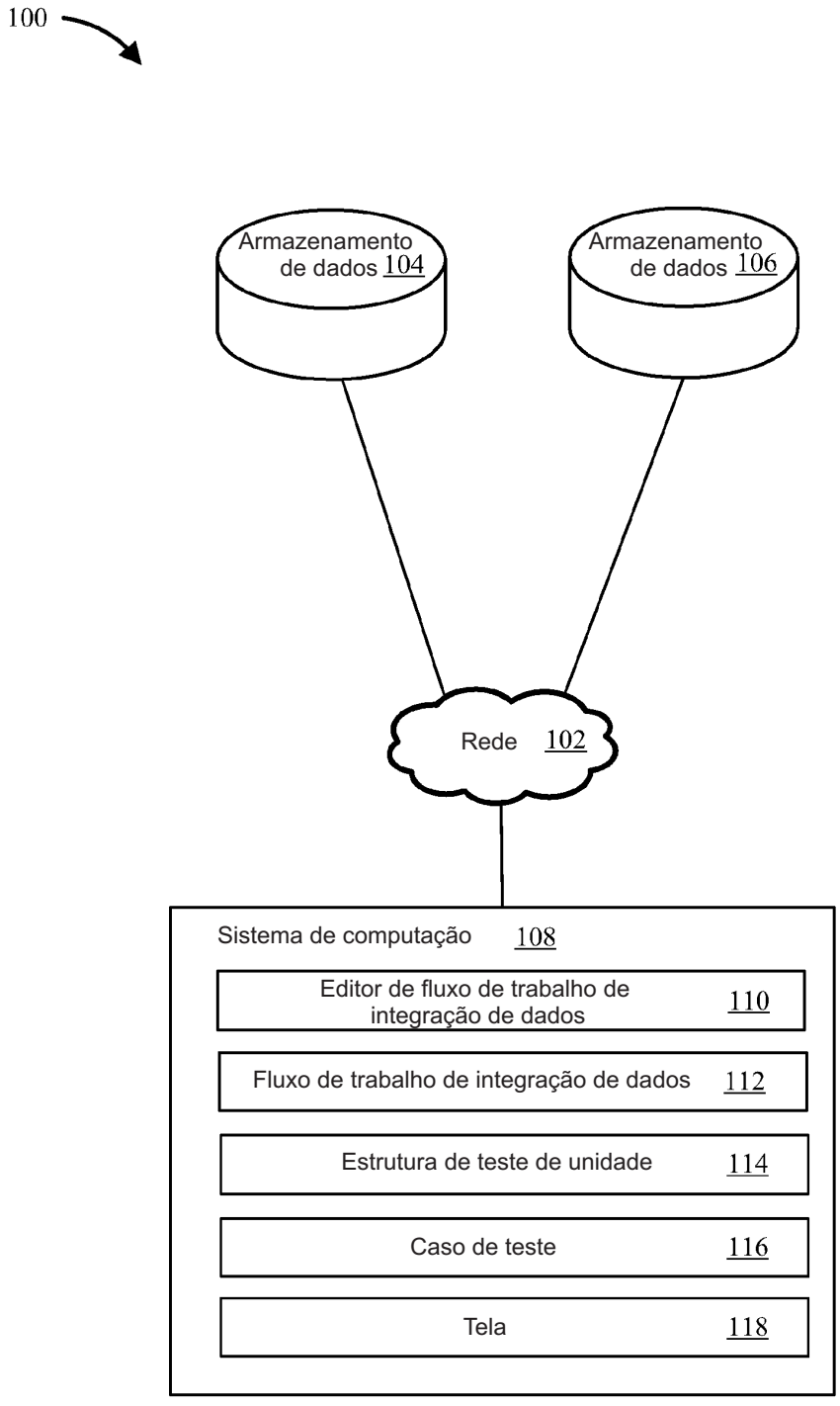
100
Armazenamento de dados 104
Armazenamento de dados 106
Rede 102
Sistema de computação 108
Editor de fluxo de trabalho de integração de dados 110
Fluxo de trabalho de integração de dados 112
Estrutura de teste de unidade 114
Caso de teste 116
Tela 118

FIG. 1

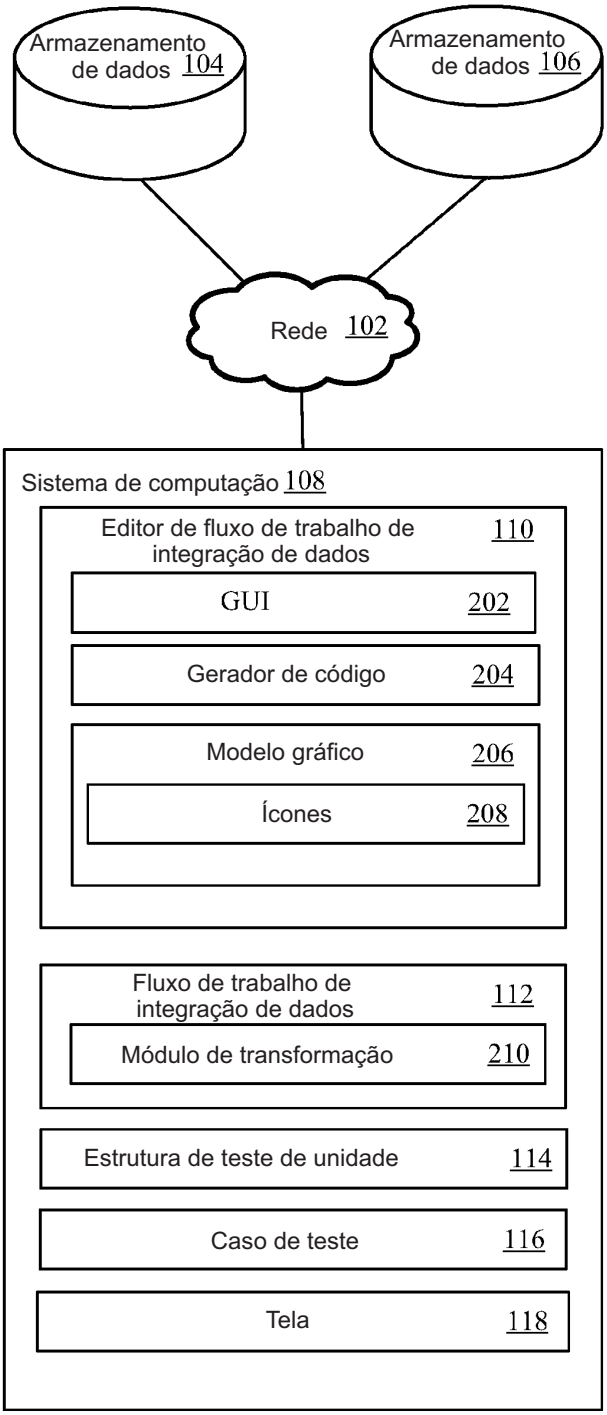
Armazenamento de dados 104
Armazenamento de dados 106
Rede 102
Sistema de computação 108
Editor de fluxo de trabalho de integração de dados 110
GUI 202
Gerador de código 204
Modelo gráfico 206
Ícones 208
Fluxo de trabalho de integração de dados 112
Módulo de transformação 210
Estrutura de teste de unidade 114
Caso de teste 116
Tela 118

FIG. 2

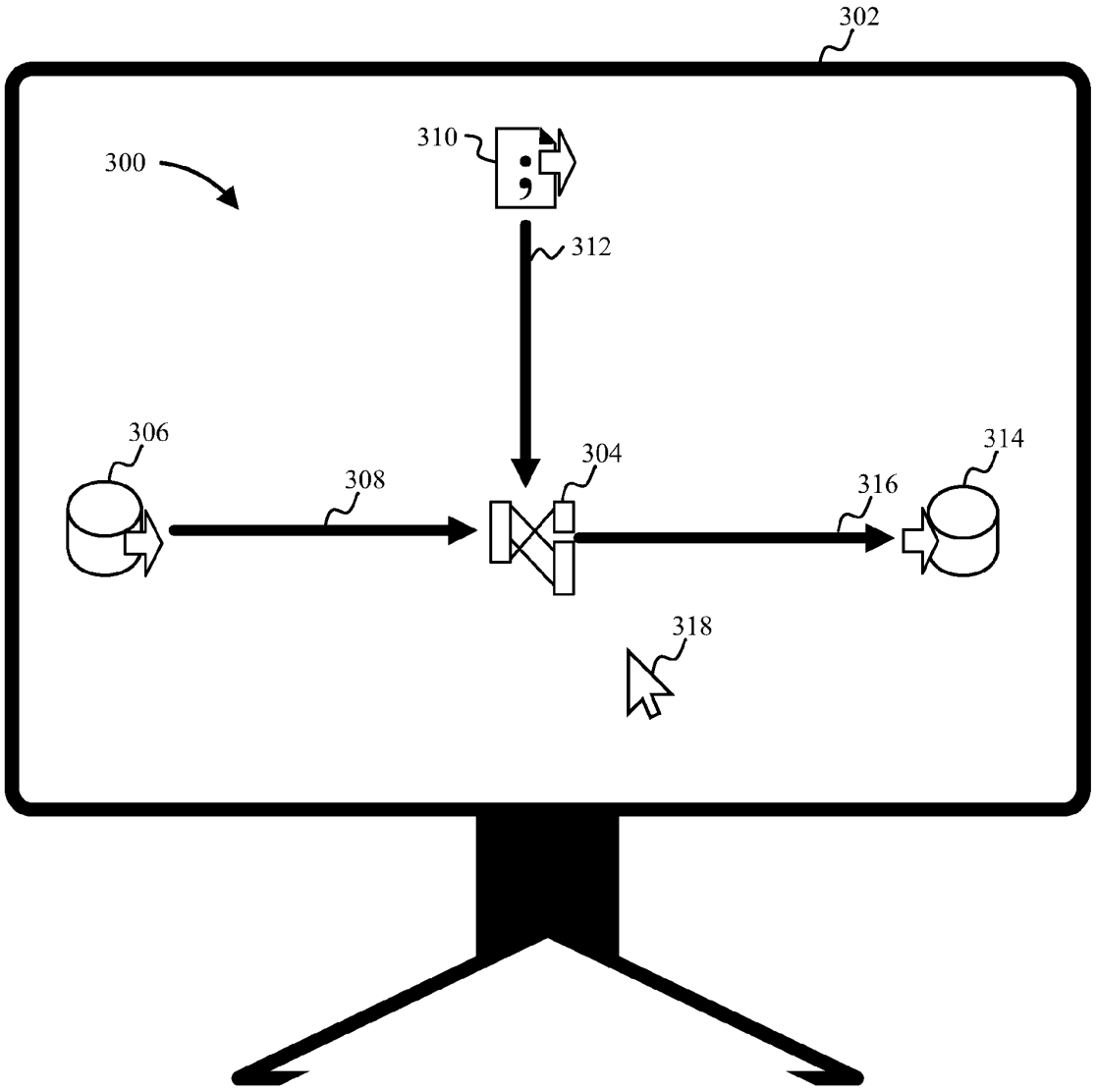

FIG. 3

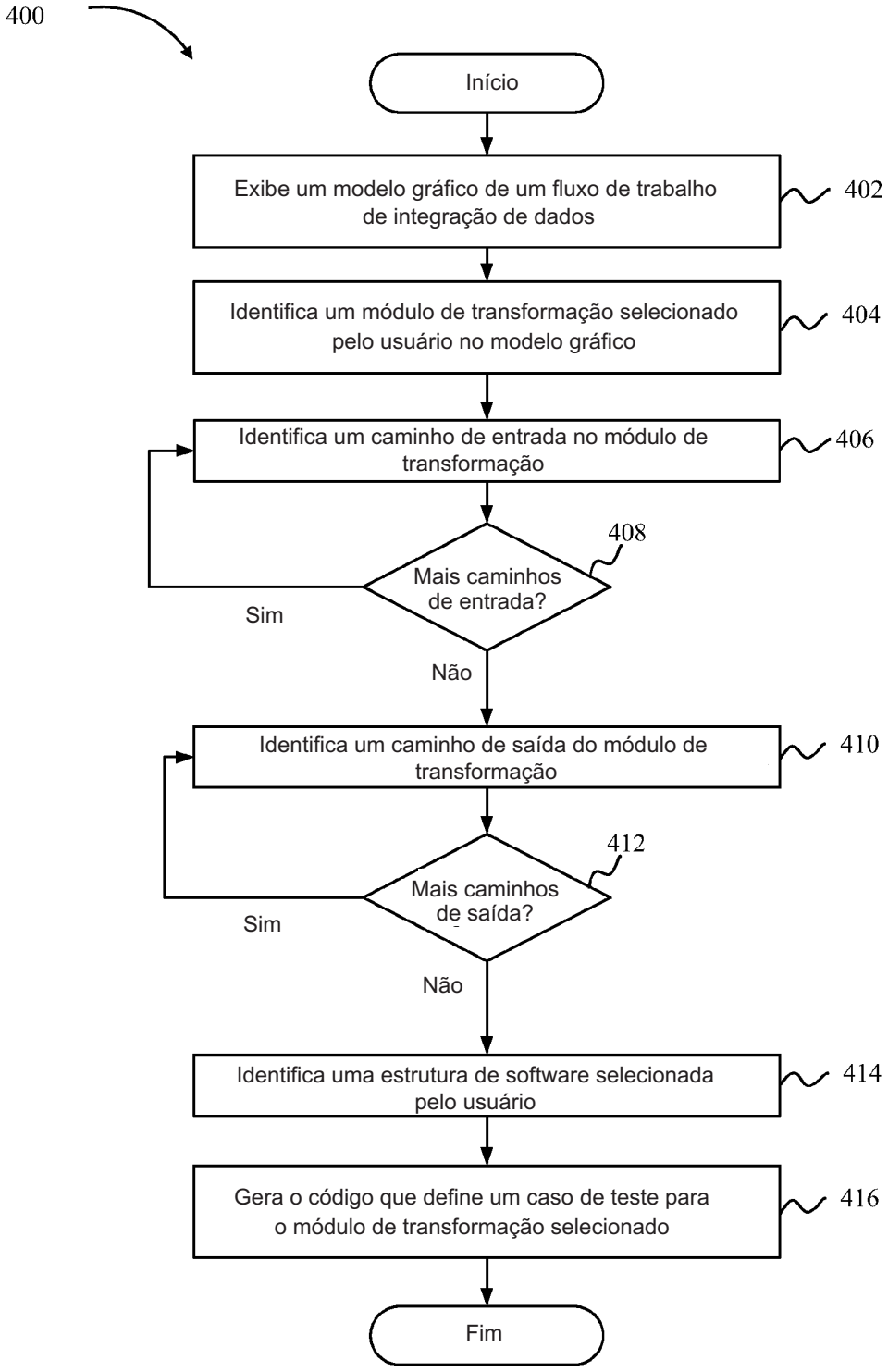
400
Início
Exibe um modelo gráfico de um fluxo de trabalho de integração de dados
402
Identifica um módulo de transformação selecionado pelo usuário no modelo gráfico
404
Identifica um caminho de entrada no módulo de transformação
406
408
Mais caminhos de entrada?
Sim
Não
Identifica um caminho de saída do módulo de transformação
410
412
Mais caminhos de saída?
Sim
Não
Identifica uma estrutura de software selecionada pelo usuário
414
Gera o código que define um caso de teste para o módulo de transformação selecionado
416
Fim

FIG. 4

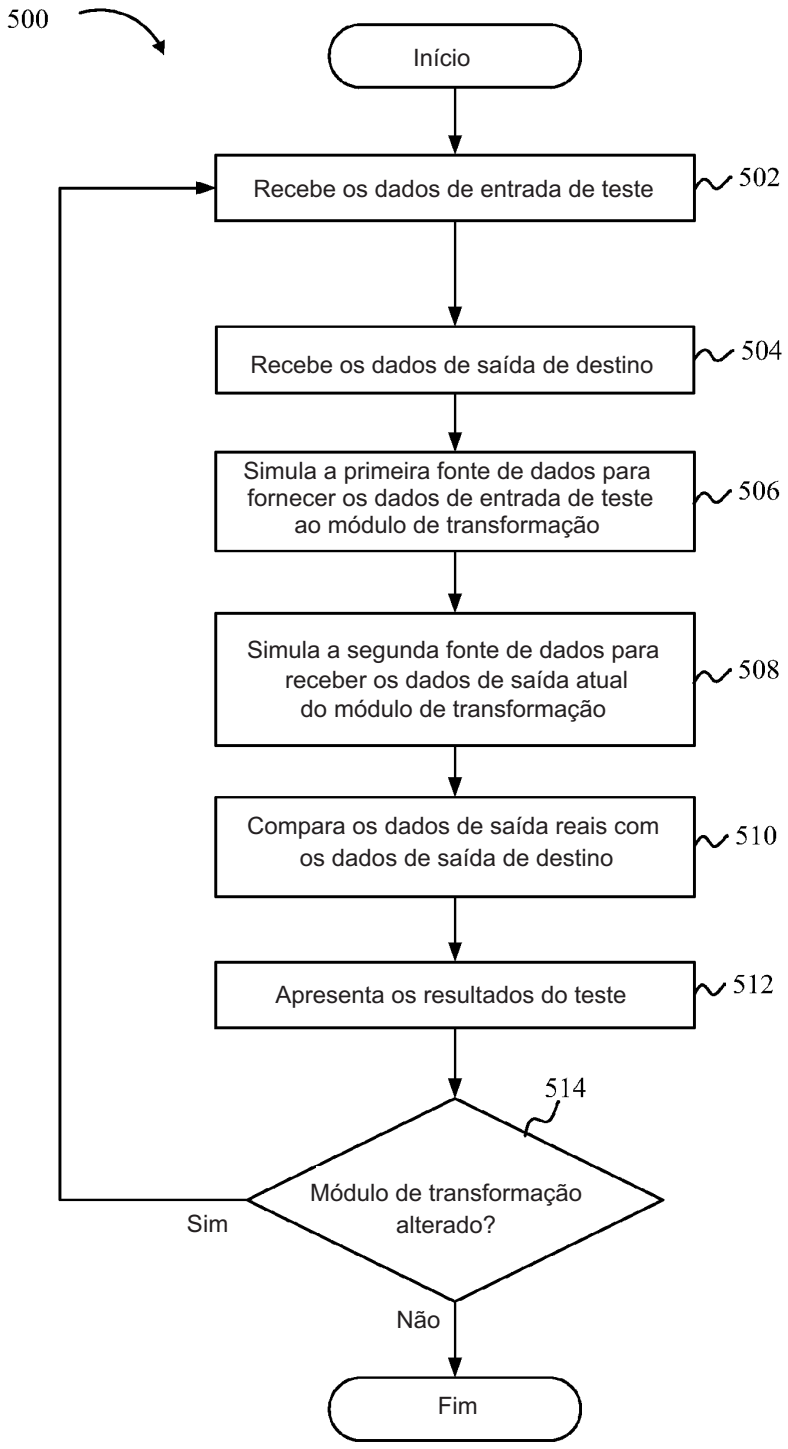
500
Início
Recebe os dados de entrada de teste
502
Recebe os dados de saída de destino
504
Simula a primeira fonte de dados para fornecer os dados de entrada de teste ao módulo de transformação
506
Simula a segunda fonte de dados para receber os dados de saída atual do módulo de transformação
508
Compara os dados de saída reais com os dados de saída de destino
510
Apresenta os resultados do teste
512
514
Módulo de transformação alterado?
Sim
Não
Fim

FIG. 5

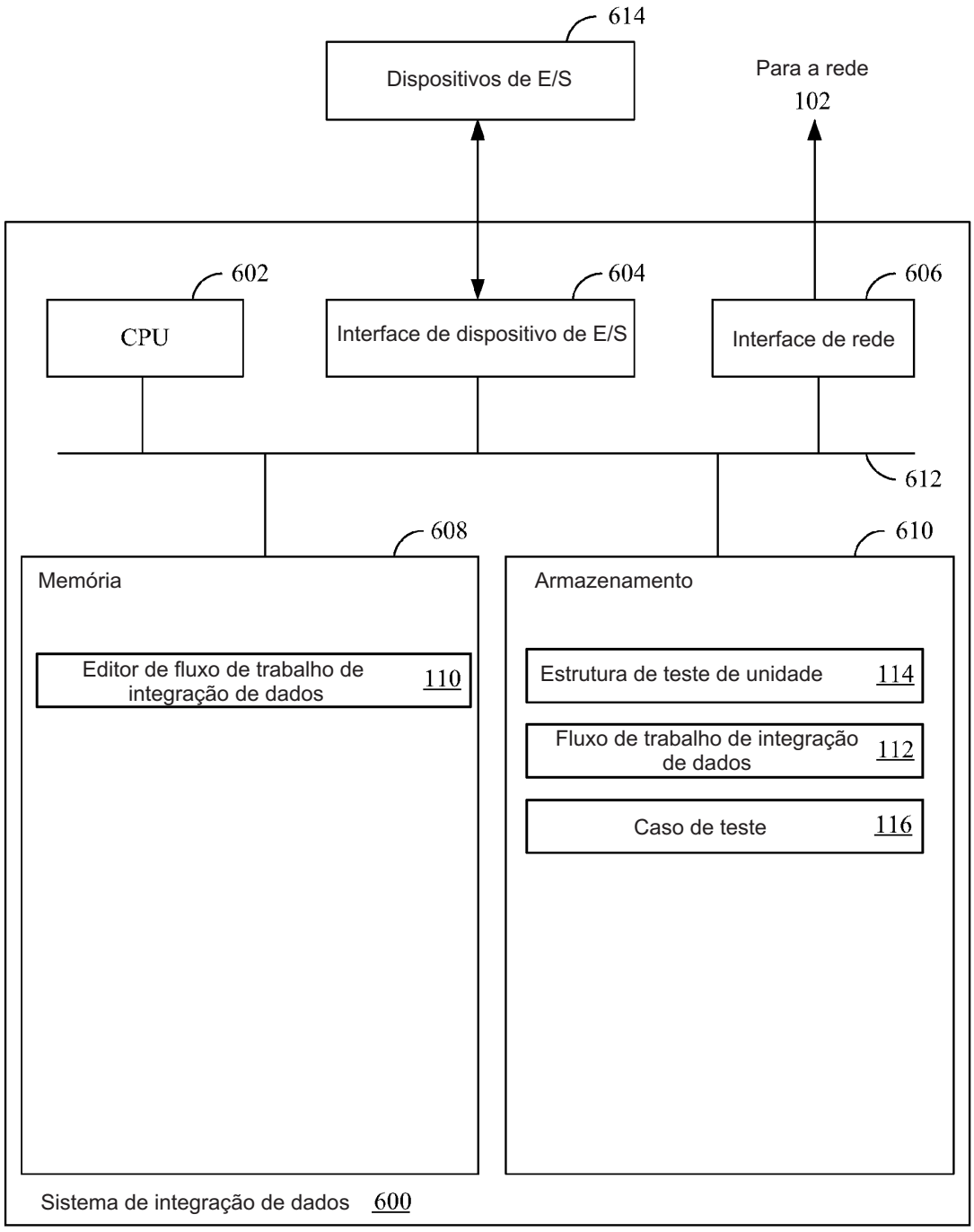
614
Dispositivos de E/S
Para a rede
102
602
CPU
604
Interface de dispositivo de E/S
606
Interface de rede
612
608
Memória
Editor de fluxo de trabalho de integração de dados 110
610
Armazenamento
Estrutura de teste de unidade 114
Fluxo de trabalho de integração de dados 112
Caso de teste 116
Sistema de integração de dados 600

FIG. 6