

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

(21) **PI0903392-0 A2**



(22) Data de Depósito: 30/04/2009
(43) Data da Publicação: 01/06/2010
(RPI 2056)

(51) *Int.Cl.:*
G06F 9/00
G06F 17/00

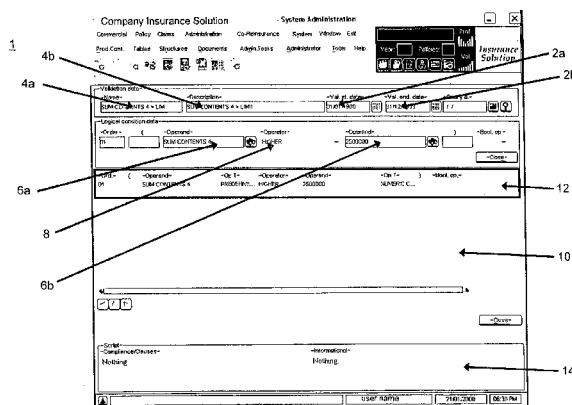
(54) Título: **MÉTODO IMPLEMENTADO EM COMPUTADOR PARA ADAPTAR E USAR UM AMBIENTE DE COMPUTADOR, PRODUTO PROGRAMA DE COMPUTADOR E SISTEMA BASEADO EM COMPUTADOR**

(30) Prioridade Unionista: 30/04/2008 EP 08 380135.7,
09/09/2008 EP 08 380266.0, 09/09/2008 EP 08 380266.0, 30/04/2008
EP 08 380135.7

(73) Titular(es): Accenture Global Services GMBH

(72) Inventor(es): Albert Juanals Pastor, Jorge Nicolau Molina,
Jose Ignacio Quiroga Serret, Pedro Olmos López, Rafael López Gracia

(57) Resumo: A presente invenção refere-se a um método implementado por computador para adaptar e utilizar um ambiente de computação, em que o ambiente de computação compreende um conjunto de arquivos executáveis, uma estrutura repositória, uma estrutura de tabela, em que a estrutura repositória e a estrutura de tabela são associadas uma à outra, e uma interface gráfica de usuário (GUI 101) que é adaptada para iniciar a execução dos arquivos executáveis, a leitura de dados e/ou escrita de dados na estrutura de tabela, e proporcionar o dado de saída como texto legível por humano, o método implementado por computador que tem as etapas de: proporcionar uma definição de uma alteração do ambiente de computação em um lado de cliente, adaptar os arquivos executáveis, de acordo com a definição da alteração do ambiente de computação em um lado de servidor, introduzir pelo menos um elemento repositório adicional na estrutura repositória no lado de servidor, em que pelo menos um elemento repositório adicional corresponde à adaptação dos arquivos executáveis, adaptar a estrutura de tabela no lado de servidor, que corresponde ao elemento repositório adicional da estrutura repositória, e acessar a estrutura de tabela adaptada no lado de servidor através da dita GUI (101) e/ou executar um ou mais arquivos executáveis no lado de servidor através da dita GUI (101), enquanto mantém substancialmente o esboço visual da dita GUI (101), assim como, um produto de programa de computador e um sistema de computador.





PI0903392-0

Relatório Descritivo da Patente de Invenção para **"MÉTODO IMPLEMENTADO EM COMPUTADOR PARA ADAPTAR E USAR UM AMBIENTE DE COMPUTADOR, PRODUTO PROGRAMA DE COMPUTADOR E SISTEMA BASEADO EM COMPUTADOR"**.

5 Descrição

A presente invenção refere-se a um método implementado por computador para adaptar e utilizar um ambiente de computação, um produto de programa de computador e um sistema baseado em computador.

No presente, na maioria dos campos industriais, os ambientes
10 de computação são usados. Embora exista um grande número de programas de computador que se encontra disponível no mercado para uma variedade de aplicações específicas, tais como, muitas aplicações industriais e muitos campos de negócios, as companhias também contam com ambientes de computação personalizados. Tal personalização pode envolver a adaptação de um sistema convencionalmente disponível, por exemplo, através de
15 uma equipe de administração interna. Entretanto, dadas as necessidades específicas, também é possível proporcionar soluções individuais para qualquer tipo de produto que uma empresa deseja estabelecer e vender ou proporcionar soluções individuais por razões logísticas, razões de manutenção, etc. Tipicamente, uma equipe de especialistas técnicos é aplicada, a fim de
20 proporcionar tais soluções personalizadas. Além disso, quando a empresa deseja explorar campos de negócios adicionais, em um cenário convencional, um ambiente de computação personalizado adicional será tipicamente estabelecido, a fim de também proporcionar produtos em tal novo campo do
25 mercado.

Um objetivo da presente invenção é aperfeiçoar a flexibilidade, com a qual um ambiente de computação pode ser proporcionado refletindo os requerimentos individuais do usuário de tal ambiente de computação.

O dito objetivo é solucionado de acordo com as reivindicações
30 independentes. As modalidades preferidas são submetidas às reivindicações dependentes.

Método implementado por computador para adaptar e utilizar um

ambiente de computação

Um aspecto da presente invenção refere-se a um método implementado por computador para adaptar e utilizar um ambiente de computação, em que o ambiente de computação compreende

5 um conjunto de arquivos executáveis,
 uma estrutura repositória,
 uma estrutura de tabela, em que
 a estrutura repositória e a estrutura de tabela são associadas umas às outras, e

10 uma interface gráfica de usuário (GUI) que é adaptada para iniciar a execução dos arquivos executáveis,

 a leitura de dados e/ou escrita de dados na estrutura de tabela, e proporcionar os dados de saída como texto legível por humano, em que o método implementado por computador tem as etapas

15 de: proporcionar uma definição de uma alteração do ambiente de computação em um lado de cliente,

 adaptar os arquivos executáveis de acordo com a definição da alteração do ambiente de computação em um lado de servidor,

20 introduzir pelo menos um elemento repositório adicional na estrutura repositória no lado de servidor, em que pelo menos um elemento repositório adicional corresponde à adaptação dos arquivos executáveis,

 adaptar a estrutura de tabela no lado de servidor, que corresponde ao elemento repositório adicional da estrutura repositória, e

25 acessar a estrutura de tabela adaptada no lado de servidor através da dita GUI e/ou executar um ou mais dos arquivos executáveis no lado de servidor através da dita GUI, enquanto mantém substancialmente o esboço visual da dita GUI.

30 O termo "estrutura repositória", conforme usado no presente pedido pode compreender um ou mais modelos e/ou um ou mais banco de dados. Em particular, a estrutura repositória pode compreender, ainda mais particularmente, consistir em um ou mais modelos e/ou um ou mais bancos de dados. Os modelos devem ser organizados em um ou mais bancos de

dados. Um "banco de dados" pode ser um banco de dados de hardware. Também é possível, que o banco de dados seja um banco de dados de software ao definir, por exemplo, uma partição, etc. de um espaço de memória existente.

5 O termo "associado", conforme usado no presente pedido pode compreender um ou mais elementos da estrutura repositória, que também podem ser referidos como elementos repositórios, podem ser correlacionados a um ou mais elementos da estrutura de tabela, que também podem ser referidos como elementos de tabela. Em particular, para cada elemento repositório pode-se proporcionar um elemento de tabela correlacionado. Como 10 um exemplo, na estrutura repositória, através dos elementos repositórios, um ou mais parâmetros/variáveis, conforme usados pelos arquivos executáveis podem ser definidos. Na estrutura de tabela, através dos respectivos elementos de tabela, os valores específicos dos ditos elementos repositórios podem ser definidos. A seguir, os elementos repositórios podem apontar para 15 um ou mais elementos de tabela. Além disso, podem existir apontadores específicos que conectam um ou mais elementos repositórios a um ou mais elementos de tabela, em particular, que podem conectar cada um dos elementos repositórios a um respectivo elemento de tabela. Em outras palavras, 20 o número de elementos repositórios e o número de elementos de tabela podem ser idênticos.

O termo "texto legível por humano", conforme contido no presente pedido pode compreender texto escrito, tais como, políticas de contrato, em particular, apólices de seguro, etc. O dito texto legível por humano pode 25 ser impresso em uma impressora e/ou proporcionado como um arquivo de dados, tal como, um arquivo PDF, um arquivo Microsoft Word[□], etc., que podem ser visualizados em um dispositivo de exibição, tal como, uma tela de computador, um PDA, etc.

O termo "em uma definição de uma alteração de um ambiente de 30 computação no lado de cliente", pode compreender a definição de um novo produto para o usuário do método implementado por computador. Ademais, o dito termo "proporcionar uma definição de uma alteração ..." pode compre-

ender a utilização de uma definição pendente de um ambiente de computação e a alteração de elementos/parâmetros específicos do dito ambiente. Como um exemplo, parâmetros/variáveis adicionais e/ou novos podem ser introduzidos. Também é possível que os parâmetros/variáveis existentes sejam alterados. Também é possível que os parâmetros/variáveis existentes sejam multiplicados. O anterior também pode compreender que o código legível por máquina adicional é proporcionado. De maneira alternativa ou adicional, o código já existente pode ser alterado.

Levando em conta todas as ditas alterações, os arquivos executáveis podem ser adaptados no lado de servidor. A adaptação dos arquivos executáveis pode compreender um ou mais arquivos executáveis, em particular, um conjunto de arquivos executáveis, ainda mais particularmente, todos os arquivos executáveis são novamente criados. A criação dos arquivos executáveis adaptados/novos pode ser realizada compilando-se o código legível por máquina adaptado no lado de servidor. Em outras palavras, proporcionando-se uma definição de uma alteração do ambiente de computação no lado de cliente, os arquivos executáveis adaptados/novos podem ser criados no lado de servidor.

De acordo com a adaptação dos arquivos executáveis, pelo menos um elemento repositório adicional é introduzido na estrutura repositória no lado de servidor. Em outras palavras, os arquivos executáveis podem ser alterados de uma maneira que um ou mais parâmetros adicionais, em particular, uma ou mais variáveis adicionais podem ser manipuladas usando os ditos arquivos executáveis. Como uma consequência, um ou mais elementos são introduzidos na estrutura repositória, que reflete os ditos parâmetros/variáveis adicionais. Os ditos novos elementos na estrutura repositória pode compreender a definição específica dos ditos novos parâmetros/variáveis.

Alterações similares podem ser realizadas na estrutura de tabela. Isto significa que, uma ou mais tabelas adicionais podem ser introduzidas. O termo "adaptar a estrutura de tabela" pode compreender a manutenção da definição de tabela geral, porém, adiciona elementos adicionais nas tabelas.

Portanto, também pode ser possível que as tabelas existentes sejam ampliadas, em particular, ao introduzir ou permitir a introdução de valores que refletem os elementos recentemente adicionados à estrutura repositória. Em outras palavras, a estrutura repositória pode compreender um ou mais parâmetros adicionais. Portanto, como um exemplo, a estrutura de tabela é modificada para permitir o armazenamento e/ou leitura de valores específicos dos ditos novos parâmetros/variáveis.

"Manter o esboço visual da dita GUI" pode significar que a maioria dos elementos da GUI é inalterada, enquanto apenas um número menor de elementos da GUI é modificado considerando, deste modo, a alteração dos arquivos executáveis e/ou a alteração dos elementos repositórios, isto é, a alteração da estrutura repositória e/ou a alteração dos elementos de tabela, isto é, a alteração da estrutura de tabela.

Em outras palavras, embora um ou mais dos elementos da GUI possam ser modificados em relação à adaptação dos arquivos executáveis e/ou à adaptação da estrutura repositória e/ou à adaptação da estrutura de tabela, no entanto, a aparência cognitiva da GUI, conforme entendida pelo usuário, pode ser substancialmente a mesma.

Consequentemente, o método implementado por computador acima permite um uso muito intuitivo e de fácil entendimento do ambiente de computação pelo usuário, mesmo quando o ambiente de computação tiver sido adaptado. Isto significa, o ambiente de computação pode ser alterado, de modo que um ou mais produtos adicionais possam ser incluídos e/ou possam ser criados ao usar a dita GUI. Isto pode compreender a obtenção de um texto legível por humano, que não foi capaz de ser obtidos antes da adaptação do ambiente de computação. No entanto, o uso do ambiente de computação se mantém substancialmente idêntico quando comparado ao uso antes da adaptação do ambiente de computação.

Adicionalmente, o método descrito acima permite um uso e/ou adaptação eficiente do ambiente de computação, enquanto conserva particularmente o uso de recursos. Este se encontra entre outros casos, uma vez que uma estrutura já existente, em particular, uma estrutura repositória e/ou

estrutura de tabela já existente pode ser facilmente modificada, de acordo com as necessidades do ambiente de computação adaptado, sem proporcionar substancialmente uma estrutura repositória e/ou estrutura de tabela completamente nova. De preferência, a estrutura repositória existente pode ser usada e pode ser apenas adicionalmente aumentada, por exemplo, adicionando-se elementos adicionais. Consequentemente, a estrutura de tabela existente também pode ser usada e pode ser apenas aumentada. Portanto, a flexibilidade do ambiente de computação é aumentada, uma vez que a mesma é facilmente adaptável às diferentes necessidades do usuário.

10 Como uma vantagem adicional, de acordo com o método descrito acima, um ambiente de computação existente pode ser ampliado de uma maneira simples e intuitiva, em um período de tempo muito curto.

15 No método, de acordo com o presente pedido, a adaptação dos arquivos executáveis pode ser realizada de maneira similar à definição e à criação dos arquivos executáveis, conforme descrito em detalhes adicionais abaixo.

20 No método, de acordo com o presente pedido, a adaptação da estrutura de tabela no lado de servidor e o acesso da estrutura de tabela no lado de servidor através da dita GUI podem ser realizados de maneira simultânea. Em outras palavras, a estrutura repositória pode ser modificada e/ou uma estrutura repositória modificada pode ser proporcionada, mediante a criação dos arquivos executáveis modificados e/ou adicionais no lado de servidor. A estrutura de tabela modificada, que pode compreender a introdução de elementos de tabela adicionais na estrutura de tabela existente e/ou
25 remoção de elementos de tabela existentes da estrutura de tabela existente e/ou criação de novas tabelas na estrutura de tabela existente pode ser realizada mediante a execução de um ou mais arquivos executáveis usando-se a GUI.

30 Com relação a isto, a GUI pode ser executada em um cliente adicional do servidor. Isto significa que se pode proporcionar uma estrutura de servidor de cliente muito complexa que compreende um dispositivo de servidor e pelo menos dois dispositivos de cliente, em que em um dispositivo

de cliente, a definição da alteração dos arquivos executáveis é realizada, no dispositivo de servidor os arquivos executáveis são alterados e no dispositivo de cliente adicional, a GUI pode ser executada. Também é possível que pelo menos uma parte ou todos os arquivos executáveis sejam executados em um cliente que também executa a GUI. Também pode ser possível compreender clientes adicionais e diversos arquivos executáveis podem ser executados em diversos clientes enquanto a GUI é executada em um cliente. De maneira similar, pelo menos um arquivo ou arquivos executáveis adicionais pode ser executado no servidor e/ou um primeiro cliente e/ou um segundo cliente e/ou um terceiro cliente, etc. A GUI pode ser executada no servidor ou em um dos clientes.

O termo "adaptado para iniciar" pode compreender que a GUI que pode ser conectada a uma funcional ou casos que podem realizar a tarefa iniciada pela GUI. Como um exemplo, a GUI pode acionar a execução de um arquivo executável selecionando-se um botão ou similar na GUI. Portanto, a GUI é adaptada para iniciar a execução de um arquivo executável.

O método acima também pode ser referido como administração de apólice.

Modalidades específicas do método implementado por computador

De maneira exemplificativa, os arquivos executáveis e a dita GUI acessam a mesma estrutura de tabela, em que o acesso compreende escrever dados e/ou ler dados a partir da mesma estrutura de tabela.

De maneira vantajosa, não é necessário que os dados sejam enviados e recebidos entre as entidades específicas, tal como um servidor de arquivo e um servidor de terminal. De preferência, os arquivos executáveis podem acessar de maneira simultânea ou sucessiva um subconjunto ou todas as tabelas da estrutura de tabelas. O acesso nesse sentido pode compreender a leitura e/ou escrita de dados na e/ou a partir da estrutura de tabela. Como um exemplo, a GUI pode ter um campo para lançar dados. Os ditos dados podem ser escritos em uma tabela específica. Além disso, a GUI pode ter um campo, tal como um botão para acionar o recibo de dados que pode ser exibido na GUI ou que pode ser introduzido em um produto de saída, tal

como, uma apólice de seguro ou similar. Portanto, os dados podem ser lidos a partir da estrutura de tabela através do uso da dita GUI, em particular, ao executar um arquivo executável, usando a dita GUI. A execução do arquivo executável pode ser acionada através da seleção do dito botão. De maneira adicional ou alternativa, a GUI pode ler os dados diretamente a partir de uma ou mais tabelas, de modo que a informação possa ser exibida na GUI em uma tela de computador para informação do usuário da GUI.

De maneira exemplificativa, a adaptação do ambiente de computação é substancialmente livre da cópia da estrutura repositória e/ou da estrutura de tabela.

Vantajosamente, de uma maneira muito rápida e eficiente, os produtos adicionais podem ser introduzidos no ambiente de computação existente. Em outras palavras, o ambiente de computação existente pode, de uma maneira muito eficiente, ser aumentado ao adicionar e/ou deletar os elementos repositórios e/ou elementos de tabela na dada estrutura repositória e/ou estrutura de tabela.

Em um método convencional, quando uma empresa deseja explorar um novo campo de negócio, todo o ambiente de computação geralmente é duplicado e a duplicata do ambiente de computação é modificada no novo campo de negócio. Em essência, isto significa, de maneira convencional, que "dois" ambientes de computação são proporcionados. De acordo com o método descrito acima, por outro lado, o ambiente existente pode ser ampliado de uma maneira simples e eficiente, em particular, para proporcionar e/ou ser adaptado aos produtos que não foram incluídos antes. Além disso, de maneira vantajosa, uma vez que o ambiente existente é ampliado, o recurso de computação, em particular, o espaço de armazenamento é conservado, ao evitar a duplicata das estruturas de banco de dados que inclui, em particular, as estruturas repositórias e/ou estruturas de tabelas.

De maneira exemplificativa, quando adapta o ambiente de computação, a estrutura repositória é substancialmente mantida.

Na descrição acima, o termo "estrutura" significa basicamente a definição geral do repositório. Isto pode compreender inúmeros modelos e/ou

como um ou mais subconjuntos de inúmeros modelos interagem e/ou são conectados uns aos outros. Além disso, também pode compreender como um ou mais subconjuntos dos modelos pode ser conectado a um ou mais subconjuntos da estrutura de tabela, por exemplo, uma ou mais tabelas.

- 5 Consequentemente, a estrutura repositória geral é substancialmente mantida, isto significa que a definição dos repositórios e/ou do número de repositórios é substancialmente inalterada. Entretanto, um ou mais elementos repositórios podem ser adicionados à estrutura repositória e/ou um ou mais elementos repositórios podem ser deletados da estrutura repositória. Em outras
- 10 palavras, um modelo adicional pode ser introduzido. Consequentemente, pode ser possível que uma linha adicional em um banco de dados seja introduzida, compreendendo um ou mais valores do dito modelo. Também é possível que um ou mais parâmetros e/ou variáveis de um modelo possam ser alterados e/ou introduzidos. Consequentemente, em uma ou mais tabelas da
- 15 estrutura de tabela, os valores específicos e/ou marcadores de posição podem ser proporcionados, de modo que os valores específicos dos ditos parâmetros/variáveis adicionais possam ser armazenados.

De maneira exemplificativa,

- usando a dita GUI, o dado de saída é gerado pelo ambiente de
- 20 computação,

e

- após a adaptação do ambiente de computação, usando a dita GUI, o dado de saída adicional é gerado pelo dito ambiente de computação adaptado.

- 25 De maneira vantajosa, após a adaptação do ambiente de computação, o dado de saída pode ser gerado, esta geração não era possível antes da adaptação. Como um exemplo, antes da adaptação, o ambiente de computação pode ser capaz de computar o dado de saída que refere-se a um ou mais produtos, tais como, uma apólice de seguro de carro e uma apólice de seguro imobiliário. Após a adaptação, o ambiente de computação po-
- 30 de gerar e produzir, adicionalmente, como um exemplo, uma apólice de seguro de saúde.

De maneira convencional, a fim de adaptar um ambiente de computação existente que pode proporcionar apólices de seguro de carro e apólices de seguro de saúde, o ambiente de computação existente precisa ser duplicado e adaptado para a apólice de seguro de saúde. Além disso, um ambiente de computação isolado precisa ser proporcionado para criar as apólices de seguro de carro e um ambiente de computação isolado foi proporcionado para a criação das apólices de seguro imobiliário. O ambiente de computação isolado, neste sentido, significa que cada ambiente de computação compreende seu próprio conjunto de arquivos executáveis, sua própria estrutura repositória, sua própria estrutura de tabela e sua própria interface gráfica de usuário.

De maneira vantajosa, tais ambientes de computação isolados podem ser evitados de acordo com o método descrito acima. De preferência, o ambiente existente pode ser modificado para usar os arquivos executáveis existentes, se necessário, modificados, a estrutura repositória existente, se necessário, modificada, a estrutura de tabela existente, se necessário, modificada e a GUI existente, se necessário, modificada, porém, ainda tem substancialmente o esboço visual idêntico.

De maneira exemplificativa, o método compreende adicionalmente a etapa de:

- adaptar a dita GUI, de acordo com pelo menos um elemento repositório adicional na estrutura repositória introduzindo-se pelo menos um elemento de exibição em uma entidade de entrada/saída dinâmica da dita GUI, enquanto mantém as entidades de entrada/saída da dita GUI substancialmente inalteradas.

De maneira vantajosa, de acordo com o método acima, proporcionar a definição de uma alteração do ambiente de computação pode envolver a adição de um ou mais elementos repositórios. A adição dos ditos elementos repositórios pode ser refletida através da adição dos elementos de exibição na GUI, em particular, em uma janela específica da GUI. As janelas, botões restantes, etc. da GUI podem manter-se substancialmente não modificados. Em outras palavras, uma entidade de entrada/saída exemplifi-

cativa pode ser uma janela da GUI.

De maneira exemplificativa, introduzindo-se pelo menos um elemento de exibição adicional na dita entidade de entrada/saída dinâmica da dita GUI, o tamanho da dita entidade de entrada/saída dinâmica é mantido.

- 5 De maneira exemplificativa, a dita entidade de entrada/saída dinâmica da GUI é implementada como uma lista suspensa, se o tamanho da dita entidade de entrada/saída dinâmica for menor que o tamanho comum de todos os elementos de exibição da dita entidade de entrada/saída.

- 10 De maneira vantajosa, no caso de o número de elementos de exibição adicionados ser muito grande, a fim de exibir todos os ditos elementos simultaneamente na dita janela, à medida que a barra de rolagem pode ser proporcionada, a fim de rolar através dos ditos elementos de exibição.

- 15 Em outras palavras, a adaptação do ambiente de computação e, em particular, a introdução de um grande número de elementos repositórios, pode ser refletida pela introdução de um grande número de elementos de exibição na GUI. É possível que para cada elemento repositório que é recentemente introduzido na estrutura repositória, um único elemento de exibição possa ser incluído na dita janela e na GUI. Embora o número de elementos possa ser muito grande, a fim de ser exibido simultaneamente na dita janela
20 através da barra de rolagem, todos os elementos podem ser exibidos para o usuário.

De maneira exemplificativa,

- para exibir um elemento de exibição na entidade de entrada/saída dinâmica, o conteúdo do dito elemento de exibição é lido a partir de
25 um elemento de tabela associado

e/ou

- para armazenar o conteúdo de um elemento de exibição da entidade de entrada/saída dinâmica, o conteúdo do dito elemento de exibição é escrito em um elemento de tabela associado.

- 30 De maneira exemplificativa,

- os elementos de tabela armazenados na dita estrutura de tabela são armazenados em uma ordem de tabela predeterminada,

em que

- os elementos de exibição exibidos na entidade de entrada/saída dinâmica são exibidos em uma ordem de exibição predeterminada,
e em que

5 - a ordem de tabela predeterminada é diferente da ordem de exibição predeterminada.

Em outras palavras, na estrutura de tabela os elementos de tabela podem ser incluídos em uma ou mais tabelas da estrutura de tabela. Os elementos de tabela podem ser armazenados em uma determinada ordem,
10 ou seja, a ordem de tabela. As informações associadas a cada um dos elementos de tabela podem ser exibidas na janela descrita acima da GUI como os elementos de exibição. A seguir, os mesmos dados podem ser referidos por nomes diferentes, ou seja, pelo nome de um elemento de tabela e pelo nome de um elemento de exibição.

15 Além disso, a ordem na qual o dado é exibido como os elementos de exibição na GUI (isto é, a ordem de exibição) pode diferir da ordem na qual o dito dado é armazenado como o elemento de tabela nas ditas tabelas (isto é, a ordem de tabela). De maneira exemplificativa, a estrutura de tabela pode compreender elementos de tabela N , em que N é um número natural. A
20 janela pode exibir elementos de exibição M , em que M é um número natural. N e M podem ser idênticos. N e M também podem diferir uns dos outros. Como um exemplo adicional, N pode ser idêntico a M , isto é, pode existir um número idêntico de elementos de tabela e elementos de exibição. Entretanto, a ordem de pelo menos dois elementos de tabela na estrutura de tabela po-
25 de ser invertida quando comparada com a ordem dos respectivos elementos de exibição, à medida que exibidos na GUI. Isto já pode ser considerado como uma ordem diferente.

De maneira exemplificativa, um dispositivo de ordem de elemento é usado

30 - para ler o conteúdo de um elemento de tabela no elemento de exibição associado na ordem de exibição,

e

- para escrever o conteúdo de um elemento de exibição no elemento de tabela associado na ordem de tabela.

De maneira vantajosa, apesar da ordem diferente dos elementos de tabela na estrutura de tabela e dos elementos de exibição, conforme exibidos na GUI usando o dispositivo de ordem, a informação é proporcionada para o usuário do ambiente de computação na forma/ordem correta. Como um exemplo, o dispositivo de ordem pode ser uma Ferramenta JAVA. Como um exemplo, a Ferramenta Java pode ser o software que suporta a extremidade frontal com o usuário. A mesma pode permitir que o usuário interaja com os processos de negócios codificados e situados no computador central. Além disso, o mesmo pode suportar a interface com o usuário ao longo das telas, e o dado lançado é processado pelas Classes Java. As classes podem ser uma ou mais faixas de campos ordenados com um certo critério, que envia ou recebe dados específicos para os ditos campos. A Ferramenta/Software Java pode usar as classes para comunicar e/ou processar os dados lançados pelo usuário.

A descrição acima refere-se, em particular, à ordem dos elementos de tabela e os elementos de exibição se aplicam, de maneira similar, à ordem dos elementos repositórios e dos elementos de tabela, assim como a ordem dos elementos repositórios e dos elementos de exibição.

Produto de programa de computador de acordo com um aspecto

Um aspecto da presente invenção refere-se a um produto de programa de computador, armazenado em um meio legível por computador ou proporcionado como um sinal de dados, que compreende instruções legíveis por computador, que quando carregadas na memória de um computador e executadas pelo computador obtém este computador que realiza o método de acordo com a invenção.

Sistema baseado em computador para gerar dados de saída de acordo com um aspecto

Um aspecto da presente invenção refere-se a um sistema baseado em computador para gerar o dado de saída em particular, sob a forma de texto legível por humano, que compreende

- um conjunto de arquivos executáveis,
 - uma estrutura repositória e
 - uma estrutura de tabela, em que
 - a estrutura repositória e a estrutura de tabela são associadas
- 5 umas às outras,
- e
- uma interface gráfica de usuário (GUI) que é adaptada para
 - executar os arquivos executáveis,
 - ler dados a partir da e/ou escrever dados na estrutura de tabela,
- 10 la, e
- proporcionar o dado de saída como texto legível por humano,
 - o sistema baseado em computador que compreende adicionalmente:
- uma interface de cliente para definir uma alteração dos arquivos
- 15 vos executáveis em um lado de cliente,
- dispositivo de servidor para
 - adaptar os arquivos executáveis de acordo com a definição da alteração em um lado de servidor,
 - introduzir pelo menos um elemento repositório adicional na estrutura repositória no lado de servidor, em que menos um elemento repositório adicional corresponde à adaptação dos arquivos executáveis
- 20
- e
- adaptar a estrutura de tabela no lado de servidor, que corresponde ao elemento repositório adicional da estrutura repositória,
- 25 e em que
- a dita GUI é adaptada para acessar a estrutura de tabela adaptada no lado de servidor e/ou para executar um ou mais arquivos executáveis no lado de servidor, enquanto o esboço visual da dita GUI é substancialmente mantido.
- 30 Além do que foi dito acima, o presente pedido também pode descrever um método implementado por computador para gerar arquivos de computador executáveis interrelacionados em um lado de servidor que com-

preendem as etapas de:

- proporcionar em um lado de cliente a interface gráfica de usuário associada ao código legível por máquina em uma linguagem de máquina, em que a interface gráfica de usuário exibe os dados substancialmente no texto claro;
- selecionar e associar duas ou mais seções de código legível por máquina na linguagem de máquina no dito lado de cliente usando a dita interface gráfica de usuário;
- a verificação implementada por computador de uma coerência do dito código legível por máquina selecionado antes no lado de cliente, de preferência, sem compilar o dito código legível por máquina selecionado no lado de cliente e produzir o resultado da dita verificação de coerência substancialmente no texto claro, em que
- a verificação da coerência do dito código legível por máquina selecionado compreende:
 - uma verificação implementada por computador da coerência das ditas seções selecionadas do código legível por máquina e/ou
 - a verificação implementada por computador da coerência das ditas associações das ditas seções selecionadas do código legível por máquina; e
- mediante a aprovação da coerência, compila as duas ou mais seções selecionadas e associadas do código legível por máquina em um lado de servidor para criar e/ou adaptar e/ou modificar dois ou mais arquivos executáveis interrelacionados que são executáveis pelo servidor.
- De maneira vantajosa, os arquivos de computador executáveis em um lado de servidor podem ser produzidos e/ou adaptados e/ou modificados por uma pessoa, substancialmente sem quaisquer habilidades específicas em programação de computador. De preferência, usando a interface gráfica de usuário (GUI) através do texto claro, o usuário pode coletar duas ou mais seções de código legível por máquina na linguagem de máquina sem ler tal código legível por máquina ou linguagem de máquina e, em particular, sem precisar saber o código legível por máquina ou linguagem de máquina;

quina. O usuário pode escolher seções, tais como, "elementos", "validações", "textos/parágrafos "..., que são associados às seções específicas do código legível por máquina na linguagem de máquina. Como um exemplo, uma validação pode ser associada ao código legível por máquina que compreende uma avaliação, se uma variável e/ou parâmetro for maior que outra variável e/ou parâmetro. As variáveis e/ou parâmetros também podem ser especificados usando a GUI, em que o valor específico da variável e/ou parâmetro é computado quando o programa é executado.

Em outras palavras, o usuário pode criar substancialmente uma validação ao lançar duas ou mais variáveis e/ou parâmetros e lançar também o tipo de validação, por exemplo, uma operação, tal como, "maior que", "menor que", "igual a", etc. Além disso, o usuário pode lançar a consequência da validação, isto é, o que será feito quando uma condição da validação for cumprida ou a dita não for cumprida. Como um exemplo, quando uma condição é cumprida, o método implementado por computador se move para uma próxima etapa.

Em outras palavras, usando a GUI, o código legível por máquina complexo na linguagem de máquina pode ser montado, sem o usuário ter conhecimento específico do código legível por máquina ou da codificação em particular. De preferência, o usuário pode ser uma pessoa versada na técnica, tal como, um engenheiro mecânico, um economista, etc., que precisa usar os arquivos de computador executáveis e, portanto, deseja proporcionar arquivos de computador executáveis de acordo com suas necessidades. Consequentemente, de maneira vantajosa, ocorre pouca perda de informação, uma vez que o usuário não precisa explicar para um inventor a necessidade dos arquivos executáveis.

De maneira preferencialmente adicional, a verificação implementada por computador de uma coerência do dito código legível por máquina seletivo no lado de cliente antes de compilar o dito código legível por máquina selecionado permite que o usuário, de um modo muito rápido e eficiente, verifique o uso dos arquivos executáveis produzidos em uma etapa posterior. Em particular, mesmo se não conectado em um lado de servidor ou se o lado

de servidor não estiver disponível para o usuário, por qualquer motivo, o usuário já pode verificar no lado de cliente, sem produzir os arquivos executáveis, se os dois ou mais arquivos executáveis irão trabalhar de maneira adequada no lado de servidor. Esta verificação pode ser particularmente feita, mesmo se o usuário não foi versado na técnica de programação específicas, uma vez que a dita verificação da coerência é implementada por computador. Em outras palavras, no lado de cliente, o computador examina as seções selecionadas do código legível por máquina. Deste modo, o computador pode examine cada uma das seções separadamente. O computador também pode examinar as seções selecionadas em sua totalidade. De preferência, o computador pode examine qualquer inter-relação entre as seções selecionadas do código. Como um exemplo, o computador pode examinar a saída de uma seção do código que é lançado em outra seção do código, como se o código fosse compilado.

Em outras palavras, o computador pode examinar, sem compilar, se uma possível saída de uma primeira seção é adequada como a entrada de outra seção. Portanto, as associações também entre as seções do código selecionado podem ser examinadas de um modo computadorizado pelo menos parcialmente automático. Deste modo, o computador pode aplicar regras específicas, tais como, regras matemáticas e/ou lógica específica, tal como, lógica matemática, no código legível por máquina. Também, o computador pode usar regras e/ou lógica de código legível por máquina, tais como regras e/ou lógica da linguagem de codificação específica, a fim de examinar as seções selecionadas e/ou as associações entre as seções selecionadas do código.

Além disso, o computador no lado de cliente (isto é, um computador ou processador de cliente) pode examinar as seções selecionadas do código e/ou as associações para integridade. Como um exemplo, se uma seção for associada à outra seção, de modo que a saída de uma seção seja lançada em outra seção, em que a outra seção precisa, por exemplo, de uma, duas ou mais variáveis de entrada, o computador pode examinar qualquer incoerência, por exemplo, se a variável de saída de uma seção de código

go atender os requerimentos da entrada de outras seções de código. Além disso, o computador pode avaliar o comportamento de tempo de execução do código, mesmo sem compilação, por exemplo, ao usar a lógica matemática. Como um exemplo, se o código compreende uma fração e o denominador da dita fração é uma variável que pode se tornar muito pequena durante o tempo de execução devido às razões matemáticas das faixas de variável/parâmetro proporcionadas, o computador no lado de cliente pode descobrir que, como um resultado, um transbordamento de dados ou similar, irá ocorrer no caso de o arquivo executável ser executado no lado de servidor.

10 Em particular, o computador no lado de cliente pode reconhecer que um transbordamento de dados irá ocorrer, mesmo sem a criação do arquivo executável (isto é, sem compilação) e mesmo sem executar o arquivo executável. Consequentemente, o computador pode proporcionar a respectiva saída de coerência.

15 A saída de coerência pode ser uma mensagem de erro ou uma mensagem de aprovação, particularmente, em texto claro. Como um exemplo, os erros matemáticos ou similares, que podem ser proporcionados quando se verifica a coerência do código selecionado e composto na linguagem de máquina, tal como, uma linguagem de programação, pode ser traduzida em texto claro, em particular, texto legível por humano, ainda mais particularmente, de acordo com as semânticas entendíveis e legíveis por humano e regras de linguagem. Consequentemente, o usuário que não é um especialista em codificação, pode verificar se as seções de código legível por máquina, que foram selecionadas usando a GUI, podem ser sucessivamente executadas no lado de servidor, sem

25

a) a necessidade de compilar as seções selecionadas de código no lado de servidor e/ou sem

b) a necessidade do usuário de ter habilidades de programação específicas, uma vez que a GUI proporciona uma saída legível e entendível por uma pessoa que não é versada na técnica de programação.

30

Como um exemplo, um tradutor pode ser proporcionado conectado e/ou usando uma ou mais tabelas que compreendem mensagens em

lógica matemática e/ou lógica de linguagem legível por máquina, etc. que são ligadas/associadas ao texto claro que cobre o respectivo significado na linguagem/lógica legível por humano.

O termo "texto claro" refere-se a uma exibição de texto, de acordo com as semânticas humanas, gramática humana, linguagem humana, particularmente independente/diferente da linguagem legível por máquina. Como um exemplo, o texto claro pode ser proporcionado em inglês, espanhol, alemão, etc. O texto claro é, particularmente, o texto que pode ser lido e entendido por uma pessoa, que não é versada na técnica de programação, como um exemplo, sem ter habilidades específicas em qualquer linguagem de programação. O texto claro pode ser proporcionado usando também figuras, imagens, pictóricas, cores diferentes e/ou esquemas de cores, realce de palavras, em particular, exibição intermitente de palavras, etc.

O termo "código legível por máquina na linguagem de máquina" (contrário ao texto claro) pode se referir a qualquer linguagem de programação convencional que pode ser, por exemplo, COBOL, FORTRAN, C, C++, PASCAL, JAVA, etc. O código legível por máquina na linguagem de máquina pode compreender elementos, que também podem ser compreendidos por texto claro na linguagem humana. Como um exemplo, um código legível por máquina na linguagem de máquina pode compreender as palavras de uma linguagem humana, em particular, palavras na língua inglesa, tais como, "e", "se", "então", etc. Entretanto, o código legível por máquina na linguagem de máquina é baseado na lógica matemática, em particular, como uma ferramenta matemática quando comparada ao texto claro que é baseado na gramática de uma linguagem humana como sua lógica intrínseca. Portanto, o termo "linguagem de máquina" não é necessariamente limitado ao significado de um baixo nível de linguagem de programação.

O termo "verificação de coerência implementada por computador" descreve o exame implementado por computador das seções de código selecionado e/ou das associações entre as seções de código selecionado, conforme descrito acima. Em particular, as seções de código selecionado podem ser separadamente examinadas. Também é possível que os conjun-

tos de seções possam ser examinados. Também é possível que a totalidade das seções selecionadas possa ser examinada. É possível que as associações entre duas ou mais seções de código sejam examinadas. As associações podem, por exemplo, compreender que a saída de uma seção é a entrada de outra seção. As associações podem, por exemplo, compreender que duas mais seções compartilham uma ou mais seções, variáveis/parâmetros adicionais, etc.

Contrário à "verificação implementada por computador de uma coerência", o termo "compilação" descreve a tradução de texto escrito em uma linguagem legível por computador que também pode ser referida como uma linguagem fonte, em outra linguagem legível por computador que também pode ser referida como a linguagem alvo. A sequência original é geralmente chamada de código fonte e a saída pode ser chamada de código de objeto. Comumente, a saída tem uma forma adequada para processamento através de outros programas (por exemplo, um vinculador).

Através da compilação, o código fonte pode ser traduzido em um programa executável ou arquivo executável. Em particular, o código fonte de um alto nível de linguagem de programação pode ser traduzido em uma linguagem de nível inferior, tal como, por exemplo, linguagem de montagem.

O termo "aprovação da coerência" descreve que uma verificação de coerência pode ser aprovada pelo computador de cliente, em particular, automaticamente, por exemplo, se um resultado específico for obtido pela verificação implementada por computador da coerência. Se um resultado específico não for obtido, é possível que uma mensagem de erro seja produzida. No entanto, mesmo no caso de tal mensagem de erro, um usuário pode aprovar de maneira alternativa ou adicional a coerência manualmente, por exemplo, se a saída da aprovação computadorizada proporcionar um significado específico no texto claro, tal como, por exemplo, certas variáveis não foram consideradas ou não foram adequadamente consideradas pelos arquivos executáveis a serem gerados, etc.

O termo "arquivos executáveis interrelacionados" descreve que dois ou mais arquivos executáveis, quando executados, trabalham juntos,

por exemplo, compartilham recursos, variáveis/parâmetros comuns, etc. Também, o termo "arquivos executáveis interrelacionados" pode descrever que um ou mais arquivos executáveis executam um ou mais outros arquivos executáveis. Isto também é possível pelo fato de que a saída de um ou mais arquivos executáveis é/são entradas de um ou mais arquivos executáveis. Deste modo, tais arquivos são interrelacionados.

O termo uma "interface gráfica de usuário é associada ao código legível por máquina" descreve que a GUI tem uma relação com o código legível por máquina, tal como, acesso a um banco de dados, em que o código legível por máquina é armazenado nas tabelas, etc.

A seguir, de acordo com este aspecto, a geração de arquivos de computador executáveis interrelacionados no lado de servidor pode ser facilmente realizada e/ou mantida. Em particular, de maneira vantajosa, o código legível por máquina na linguagem de máquina pode ser proporcionado e composto de uma maneira muito fácil e suficiente por uma pessoa que não é versada na técnica de programação. De maneira adicionalmente vantajosa, tal composição de código legível por máquina na linguagem de máquina pode ser obtido em um tempo muito curto, uma vez que compilações tipicamente muito compridas não precisam ser necessariamente realizadas muitas vezes. De preferência, as seções selecionadas do código legível por máquina precisam ser compiladas apenas uma vez, após uma aprovação bem-sucedida da coerência. Em relação a isto, a verificação implementada por computador de uma coerência pode ser realizada de maneira iterativa, particularmente, enquanto a coerência é aprovada.

Em outras palavras, no caso de uma mensagem de erro ser criada, usando a GUI, a seleção de duas ou mais seções de código legível por máquina pode ser alterada e/ou as associações entre as duas ou mais seções de código legível por máquina podem ser alteradas. A etapa de selecionar e associar duas ou mais seções de código legível por máquina também pode incluir o ajuste de variáveis/parâmetros usados pelas ditas duas ou mais seções de código legível por máquina. Consequentemente, no caso em que nenhuma aprovação da coerência foi ainda estabelecida, em vez de

selecionar e associar as seções adicionais do código legível por máquina e/ou não selecionar e/ou não associar uma ou mais seções de código legível por máquina, as variáveis/parâmetros das duas ou mais seções selecionadas de código legível por máquina podem ser alterados, deletados, recém ajustados, etc. É possível que a etapa de alterar, deletar, reinicializar, etc., dos parâmetros/variáveis é realizada de maneira iterativa em combinação com a verificação implementada por computador de uma coerência, até a aprovação bem-sucedida da coerência.

10 A seleção e a associação de duas ou mais seções de código pode compreender, de preferência, um ou mais:

- lançar uma ou mais regras, tais como, condições, operações matemáticas, etc. usando a GUI, em que as ditas uma ou mais regras, tais como, condições, operações matemáticas, etc são lançadas em texto claro;

15 – armazenar a entrada em uma ou mais tabelas de definição específica;

- armazenar uma ou mais regras, tais como, condições, operações matemáticas, etc. em uma tabela de condições específicas, de preferência, diferente das tabelas de definição;

20 – armazenar na(s) tabela(s) de definição e/ou na(s) tabela(s) de condição um ou mais indicador(es), tais como, por exemplo, ">", "<", "=>", "<=", "=", etc., que indicam a natureza matemática de uma ou mais regras;

- ler, por exemplo, usando um criador, a partir da(s) tabela(s) de definição e/ou tabela(s) de condição uma ou mais regras e/ou um ou mais indicador(es);

25 – proporcionar, por exemplo, usando o dito criador, em uma ou mais tabela(s) de código na linguagem legível por máquina;

– gerar, por exemplo, usando o dito criador, um ou mais arquivos executáveis interrelacionados a partir do dito código legível por máquina. É possível que a partir de cada tabela de código um arquivo executável seja
30 gerado. Também é possível que a partir de uma tabela de código dois ou mais arquivos executáveis sejam gerados.

De maneira exemplificativa, selecionar e associar duas ou mais

seções de código legível por máquina na linguagem de máquina compreende lançar regras no texto claro usando a dita interface gráfica de usuário e proporcionando, por exemplo, usando o dito criador, o código legível por máquina na linguagem de máquina equivalente ao dito texto claro que usa um gerador.

O termo "equivalente" no contexto do presente pedido compreende, por exemplo, que uma regra, condição matemática, etc., é lançada no texto claro (que pode ser facilmente entendido pelo usuário que não é versado na programação em uma linguagem de programação) e é, então, transformado na linguagem legível por máquina que pode ser interpretado, por exemplo, por um computador.

De maneira exemplificativa, a etapa de a verificação implementada por computador da coerência compreende uma verificação implementada por computador da compatibilidade da primeira seção do código usado para gerar um primeiro arquivo executável com uma segunda seção de código usada para gerar um segundo arquivo executável.

Em outras palavras, de maneira vantajosa, é facilmente possível examinar, se dois arquivos executáveis estarão trabalhando juntos de maneira adequada, isto é, se a interrelação dos ditos dois arquivos executáveis for correta, sem a necessidade de o usuário ser versado na técnica de programação específicas no código legível por máquina na linguagem de máquina.

De maneira exemplificativa, a etapa de verificação implementada por computador da coerência compreende:

- baseado no código legível por máquina selecionado que avalia no lado do cliente os parâmetros de tempo de execução antes de compilar as seções selecionadas do código legível por máquina, em que os parâmetros de tempo de execução são associados aos arquivos executáveis, quando executam os arquivos executáveis no lado de servidor.

Em outras palavras, analisando-se o código legível por máquina selecionado, em particular, analisando-se a associação entre o código legível por máquina selecionado que usa, por exemplo, lógica de linguagem de programa e/ou lógica matemática, os parâmetros de tempo de execução podem

ser avaliados antes sem compilar as seções selecionadas do código legível por máquina. Os parâmetros de tempo de execução podem ser variáveis/parâmetros usados por ou no(s) arquivo(s) executável(eis) no lado de servidor. No caso de os ditos parâmetros de tempo de execução serem incompatíveis com o sistema, em particular, incompatíveis com as configurações de hardware no lado de servidor, uma respectiva mensagem pode ser gerada de maneira automática e computadorizada. Portanto, mesmo sem ser versado na técnica de programação específicas, o usuário pode verificar de uma maneira simples se o código legível por máquina selecionado irá atuar de maneira realizar de adequada no lado de servidor.

Como exemplos, a etapa de verificação implementada por computador da coerência pode compreender:

- Quando em uma lógica condicional (também referido como validação) um elemento definido como numérico é comparado com um elemento alfanumérico, proporciona-se uma mensagem de erro que informa sobre uma incompatibilidade.

- Quando um formato de um ou mais cálculos não for correspondente/compatível com o formato de seus operandos, proporciona-se uma mensagem de erro. De maneira exemplificativa, pode-se proporcionar uma mensagem de erro quando a operação X for definida como Dec(6,2), isto é, uma variável que tem 6 entradas e 2 decimais e sua operação é o resultado da operação para resumir as variáveis Y e Z, onde Y é definido como Dec(15,2) e Z como Dec(15,2). Em outras palavras, o resultado desta soma "Y+Z" pode não se encaixar no elemento definido em X como "X = Y + Z".

De maneira exemplificativa, uma ou mais seções de código legível por máquina são reutilizadas para compilar os dois ou mais arquivos executáveis interrelacionados.

Em outras palavras, as seções similares ou idênticas do código legível por máquina podem ser usadas para a compilação. A seleção das seções de código legível por máquina pode, portanto, compreender a cópia das ditas seções. Entretanto, mesmo que as seções similares ou idênticas do código legível por máquina sejam usadas para compilar os diferentes ar-

quivos executáveis, os ditos arquivos podem atuar de maneira diferente, por exemplo, no caso de variáveis/parâmetros serem ajustadas de maneira diferente.

Em outras palavras, podem-se proporcionar grupamentos de código legível por máquina, em que os ditos grupamentos podem ser unidos ao usar a GUI a fim de proporcionar um conjunto de grupamentos que pode ser compilado em um arquivo executável no lado de servidor. O dito conjunto de grupamentos pode compreender seções similares, particularmente idênticas ao código legível por máquina. Os grupamentos também podem compreender diferentes conjuntos de código legível por máquina. Em outras palavras, a GUI pode ser usada para construir o código legível por máquina para compilar dois ou mais arquivos executáveis, em que o código legível por máquina para cada arquivo executável pode ser composto similar a um kit de construção, em que o usuário usa o texto claro proporcionado e lançado pela GUI.

De maneira exemplificativa, pelo menos um arquivo executável em primeiro nível compilado que é interrelacionado com pelo menos um arquivo executável em nível i , em que pelo menos um arquivo executável em nível i é interrelacionado com pelo menos um arquivo executável em nível $(i+1)$, em que i é um número natural maior que 1.

De maneira exemplificativa, pelo menos um arquivo executável em nível i e pelo menos um arquivo executável em nível $(i+1)$ são compilados a partir de uma ou mais seções de um código legível por máquina, em que uma ou mais seções de código legível por máquina compreendem código legível por máquina similar, de preferência, substancialmente idêntico.

Em outras palavras, um conjunto hierárquico de arquivos executáveis pode ser criado, em que os arquivos executáveis podem ser associados aos arquivos executáveis que, novamente podem ser associados aos arquivos executáveis adicionais. Os conjuntos diferentes de arquivos executáveis pode ser criado, em que cada conjunto pode ter a dita estrutura de árvore/hierárquica. Cada nível pode ter um ou mais arquivos executáveis. Cada nível pode compreender arquivos executáveis, que podem ser similares, mesmo idênticos a um ou mais níveis de arquivo executável superiores

na hierarquia de um ou mais níveis inferiores na hierarquia. Os arquivos executáveis em um nível podem ser interrelacionados. Um ou mais arquivos executáveis em um nível podem ser interrelacionados a um ou mais arquivos executáveis em um ou mais níveis hierárquicos mais altos e/ou um ou mais níveis hierárquicos mais baixos. Os conjuntos diferentes de arquivos executáveis podem ter estrutura hierárquica similar, mesmo idêntica. Os diferentes conjuntos de arquivos executáveis podem ter diferentes estruturas hierárquicas. O valor de i pode ser 2, 3, 4, 5, 6, 7, 8, 9, 10, etc.

De maneira exemplificativa, o método compreende a etapa adicional de verificar a integridade da execução de todos os arquivos executáveis ao executar todos os arquivos executáveis no lado de servidor e comparar o desempenho no lado de servidor com uma especificação predeterminada no lado de cliente.

O desempenho no lado de servidor pode ser computadorizado, comparado com uma especificação predeterminada no lado de cliente. Como um exemplo, uma ou mais saídas dos arquivos executáveis no lado de servidor podem ser comparadas com um conjunto de variáveis/parâmetros no lado de cliente.

De maneira exemplificativa, as etapas no lado de cliente são realizadas no computador de cliente remoto a partir do servidor e as etapas no lado de servidor são realizadas no computador servidor remoto a partir do cliente.

Em outras palavras, as etapas no lado de cliente são realizadas em um computador de cliente que pode ser um computador em uma rede. O computador de cliente pode ser conectado a um computador servidor através da rede. Após realizar as etapas no computador de cliente, em particular, mediante a aprovação da coerência, as duas ou mais seções selecionadas e associadas a um código legível por máquina podem ser transferidas para o lado de servidor, em particular, para um computador servidor, por exemplo, através da dita rede. As duas ou mais seções de código legível por máquina selecionadas e associadas também podem ser transferidas para o servidor através de quaisquer outros meios, tal como, ao armazenar as duas ou mais

seções de código legível por máquina selecionadas e associadas em um dispositivo de armazenamento, tal como, um CD, um flash drive, um disco rígido, etc., e copiar as ditas duas ou mais seções de código legível por máquina selecionadas e associadas a partir do dito dispositivo de armazenamento no servidor. Então, no servidor, as duas ou mais seções de código legível por máquina selecionadas e associadas podem ser compiladas.

O dispositivo de armazenamento também pode ser parte de um computador intermediário, que é conectado ao computador de cliente e ao computador servidor. Em outras palavras, as duas ou mais seções de código legível por máquina selecionadas podem ser transferidas a partir do computador de cliente para o computador servidor através de outro computador ou outro dispositivo computadorizado, tal como, um dispositivo móvel, para o servidor.

De maneira alternativa, todas as etapas no lado de cliente e as etapas no lado de servidor podem ser realizadas em um computador, que incorpora o servidor e o cliente.

Além do que foi dito acima, o presente pedido pode descrever também um sistema computadorizado para gerar arquivos de computador executáveis interrelacionados em um lado de servidor que compreende:

- uma interface gráfica de usuário em um lado de cliente, em que
- a interface gráfica de usuário é associada ao código legível por máquina,
- a interface gráfica de usuário exibe dados substancialmente em texto claro
- a interface gráfica de usuário é adaptada para selecionar duas ou mais seções de código legível por máquina em um lado de cliente;
- um verificador de coerência que é adaptado para
- verificar uma coerência do dito código legível por máquina selecionado no lado de cliente e
- produzir o resultado da dita verificação de coerência substancialmente em texto claro; e
- um compilado para compilar as duas ou mais seções de código

- * legível por máquina selecionadas em um lado de servidor para criar e/ou adaptar e/ou modificar dois ou mais arquivos executáveis interrelacionados que são executáveis pelo servidor.

- Para selecionar duas ou mais seções de código legível por máquina em um lado de cliente, a GUI pode ser conectada um seletor. O seletor pode compreender uma ou mais ligações específicas que apontam para as seções de código na linguagem legível por máquina. Em outras palavras, o seletor pode ser adaptado para selecionar duas ou mais seções de código, mediante a entrada do usuário através da GUI, deste modo, o usuário pode
- 5 se comunicar com a GUI em texto claro na linguagem legível por humano e a GUI, através do seletor pode selecionar as duas ou mais seções de código que representam o texto claro no código legível por máquina. No contexto do presente pedido, o texto claro também pode compreender que os campos de exibição de GUI, em particular, botões e/ou outros tipos de menus, que são
- 10 construídos, de modo que o usuário possa entender de maneira intuitiva o significado dos ditos campos e/ou botões e/ou outros tipos de menus. Como um simples exemplo, a fim de criar uma validação, o menu pode compreender botões que têm representados nos mesmos operadores matemáticos, tais como, ">", "<", "=", que o usuário pode entender para representar as operações em uma validação de "maior", "menor" e "igual a".
- 15 20

Devido às seções de termo de código, os detalhes acima são realizados devido ao método ser igualmente aplicável.

- De maneira exemplificativa, a interface gráfica de usuário compreende um gerador que é adaptado para proporcionar o código legível por máquina na linguagem de máquina equivalente ao dito texto claro como entrada através da interface gráfica de usuário.
- 25

- De maneira exemplificativa, o sistema computadorizado pode compreender um avaliador que é adaptado para avaliar no lado de cliente baseado no código legível por máquina selecionado os parâmetros de tempo de execução antes de compilar as seções selecionadas do código legível por máquina, em que os parâmetros de tempo de execução são associados aos arquivos executáveis quando executam os arquivos executáveis no lado de
- 30

servidor.

Além do que foi dito acima, o presente pedido pode também descrever um produto de programa de computador, armazenado em um dispositivo de armazenamento de dados ou proporcionado como sinal, que quando carregado na memória de um computador e executado pelo computador faz com que o computador realize um método conforme descrito acima.

A descrição precedente dos aspectos não se limita aos respectivos aspectos. De preferência, a descrição dos respectivos aspectos pode ser aplicada nos aspectos restantes. Em particular, a descrição que refere-se ao método pode ser aplicada no sistema de computador e no produto de programa de computador e nas respectivas modalidades específicas, consequentemente, de modo que ao combinar os recursos individuais dos aspectos ou modalidades específicas, novos aspectos e/ou modalidades possam ser criados. Além disso, os aspectos e/ou modalidades descritos acima não são restritos a uma aplicação, tal como, a geração de arquivos de computador executáveis interrelacionados que podem ser produtos de seguro ou ser relacionados a produtos de seguro e/ou relacionados às aplicações de gerenciamento de reivindicação de seguro e/ou relacionados à geração de documentos de seguro e/ou relacionados à geração de produtos adicionais, tais como, produtos de bens imobiliários, produtos e/ou documentos de serviço financeiro em um ou mais dos campos mencionados acima.

Descrição das figuras

Os seguintes exemplos específicos são descritos por meio de figuras exemplificativas, em que os recursos individuais das figuras podem ser combinados com os exemplos adicionais, não explicitamente estabelecidos na descrição das figuras. Com referência às figuras, mostrou-se na:

a figura 1a é visão geral esquemática de configurações exemplificativas das seções de código legível por máquina;

a figura 1b é uma visão geral esquemática;

figura 1c é uma captura de tela de uma GUI;

figura 2 é uma visão geral detalhada esquemática de um elemento, de acordo com a figura 1a;

a figura 3 é uma captura de tela de uma GUI que compreende realce;

a figura 4 é uma captura de tela que compreende realce;

a figura 5a é uma captura de tela que compreende realce;

5 a figura 5b é uma captura de tela que compreende realce;

a figura 6 é uma captura de tela que compreende realce;

a figura 7a é uma captura de tela de uma GUI que compreende realce;

10 a figura 7b é uma captura de tela de uma GUI que compreende realce;

a figura 8a é uma captura de tela de uma GUI que compreende realce;

a figura 8b é uma a captura de tela de uma GUI que compreende realce;

15 a figura 9 é uma captura de tela que compreende realce;

a figura 10a é uma captura de tela que compreende realce;

a figura 10b é uma captura de tela que compreende realce;

a figura 11a é uma captura de tela que compreende realce;

a figura 11b é uma captura de tela que compreende realce;

20 a figura 12a é uma captura de tela que compreende realce;

a figura 12b é uma captura de tela que compreende realce;

a figura 12c é uma captura de tela que compreende realce;

a figura 12d é uma captura de tela que compreende realce

25 a figura 13: a visão geral esquemática de um elemento, de acordo com figura 1a;

a figura 14a é uma captura de tela de uma GUI que compreende realce;

a figura 14b é uma captura de tela de uma GUI que compreende realce;

30 a figura 14c é uma captura de tela de uma GUI;

a figura 14d é uma captura de tela de uma GUI;

a figura 15 é uma captura de tela que compreende realce;

- a figura 16a é uma captura de tela que compreende realce;
a figura 16b é uma captura de tela que compreende realce;
a figura 17a é uma captura de tela que compreende realce;
a figura 17b é uma captura de tela que compreende realce;
5 a figura 18a é uma captura de tela que compreende realce;
a figura 18b é uma captura de tela;
a figura 18c é uma captura de tela que compreende realce;
a figura 19 é uma visão geral esquemática de um elemento de
acordo com figura 1a;
- 10 a figura 20a é uma captura de tela;
a figura 20b é uma captura de tela;
a figura 20c é uma captura de tela;
a figura 21a é uma captura de tela;
a figura 21b é uma captura de tela;
- 15 a figura 22 é uma captura de tela;
a figura 23 é uma visão geral esquemática que mostra a execu-
ção de dois ou mais arquivos executáveis interrelacionados;
a figura 24 é um fluxograma;
- 20 a figura 25 é uma captura de tela de uma GUI;
a figura 26 é uma captura de tela de uma GUI;
a figura 27a é uma captura de tela de uma GUI;
a figura 27b é uma captura de tela de uma GUI;
a figura 27c é uma captura de tela de uma GUI;
a figura 28 é uma captura de tela de uma GUI;
- 25 a figura 29 é uma captura de tela de uma GUI;
a figura 30 é um fluxograma;
a figura 31 é uma visão geral;
a figura 32 é uma visão geral;
a figura 33 é uma visão geral;
- 30 a figura 34 é uma visão geral;
a figura 35 é uma visão geral;
a figura 36 é uma visão geral e

a figura 37 é um sistema baseado em computador.

Os exemplos específicos do método e do sistema descritos a seguir também podem ser referidos como "Fábrica de Produto Info2000" ou "módulo de Fábrica de Produto Info2000" ou "Administração de apólice". De maneira vantajosa, o módulo de Fábrica de Produto Info2000 proporciona uma metodologia de desenvolvimento comum para todas as Linhas de Negócio, em que substancialmente todos os produtos isto é, dois ou mais arquivos executáveis interrelacionados são criados, suportados e manipulados do mesmo modo, usando as mesmas telas e um dicionário de dados comum para o armazenamento de dados. De maneira adicionalmente vantajosa, o mesmo também permite uma reutilização de componente de produto extensiva, tal como, a reutilização de seções de código na linguagem legível por máquina. No contexto deste pedido, o termo "reutilizar" significa particularmente que uma seção de código específica pode ser copiada uma ou mais vezes, para utilização em uma ou mais composições de código que são, então, usadas para criar dois ou mais arquivos executáveis interrelacionados. Em outras palavras, o termo "selecionar", no contexto deste pedido, significa que a seção de código selecionada é copiada para criar dois ou mais arquivos executáveis interrelacionados. Em particular, o termo "selecionar e associar", no contexto do presente pedido pode significar que o código é copiado nas tabelas. De maneira vantajosa, a "Administração de Apólice" pode permitir a criação e/ou manutenção de dados de apólice e/ou apólices específicas de uma maneira simples e intuitiva. Em particular, um sistema comum pode ser facilmente adaptado, enquanto a interface de "Administração de Apólice" se mantém substancialmente idêntica.

Uma vez que o Produto é definido, isto é, uma vez que todas as seções de código legível por máquina foram selecionadas e associadas, possivelmente, todos os parâmetros/variáveis necessários também foram ajustados, a Fábrica de Produto irá gerar um Programa executável que compreende dois ou mais arquivos executáveis interrelacionados, chamados de interfaces Administração de Apólice padrões.

1. Estrutura de produto

A figura 1a proporciona um diagrama que mostra a estrutura de produto básica definida na Fábrica de Produto. Em outras palavras, a figura 1a mostra uma visão geral da funcionalidade dos dois ou mais arquivos executáveis interrelacionados, uma vez que eles são executados no lado de servidor.

O criador do produto irá definir os diferentes componentes que são necessários para construir o Produto, tais como,

- Objetos segurados (por exemplo, carro, casa)
- Coberturas (por exemplo, construção)
- Códigos de avaliação (para reivindicações; por exemplo, fogo)

Estes componentes podem ser reutilizados entre os produtos. Portanto, o criador do produto pode selecionar os componentes que usam uma GUI, em que na GUI os componentes que são disponíveis usam texto claro, de preferência, pelo fato de que a GUI compreende os respectivos botões, tais como, por exemplo, um botão para "objetos segurados", que podem proporcionar, por exemplo, uma lista suspensa de possíveis escolhas que, então, podem ser selecionadas, por exemplo, simplesmente através do realce que usa um dispositivo apontador como um mouse ou algo equivalente. Estes botões são associados ao respectivo código legível por máquina na linguagem de máquina.

Os elementos também podem ser definidos e designados no nível adequado (por exemplo, ano de construção). Os elementos podem ser usados para capturar dados ou ajustar regras. Os elementos podem ser simples campos de entrada ou relacionados a um conjunto de valores de referência que podem ser mantidos ou armazenados em uma tabela de corporação.

Como um exemplo, na figura 1a, o produto a ser criado refere-se ao seguro de "propriedade de conforto domiciliar". Em outras palavras, os arquivos executáveis interrelacionados irão proporcionar, quando executados no lado de servidor, como um resultado de documentos de seguro, tal como um contrato de um seguro de uma casa. O método, de acordo com o presente pedido, permite a geração computadorizada de dois ou mais arquivos e-

executáveis interrelacionados que, quando executados, criam ou auxiliam quando cria os ditos documentos de seguro.

Em outras palavras, a figura 1a mostra um ou mais dos seguintes componentes:

- 5 - O produto refere-se a duas entidades:
 - Produto Técnico: O mesmo contém as especificações técnicas. É a estrutura de referência que é usada para criar os produtos comerciais que são realmente distribuídos.

- Produtos Comerciais: Baseados nos produtos técnicos e limita-
- 10 dos a sua área de aplicação. Eles podem apresentar restrições que limitam algumas das características do produto técnico. Cada apólice usada no método/sistema descrito no presente pedido é claramente ligada a um único produto técnico-comercial.

- O tipo de objeto refere-se ao risco ou riscos específicos cobertos
- 15 pelo produto. O método, de acordo com este pedido, permite que os produtos sejam:

- Múltiplos objetos: que contêm diferentes tipos de necessidades de seguro (pessoa, automóvel, casa, etc.);

- Múltiplas situações: a fim de repetir o mesmo tipo de objeto
- 20 quantas vezes requeridas em uma apólice. Isto permite, por exemplo, o gerenciamento de apólices coletivas, de acordo com o presente pedido.

Classe Prêmio: identifica o risco segurável (garantia) para a avaliação de um prêmio. As características importantes das classes prêmio, de acordo com o presente pedido são:

- 25 - Elas são o nível mínimo da estrutura de produto para o qual os resultados econômicos são calculados (prêmios, taxas, descontos, etc.)

- Um tipo de objeto (ou necessidade de seguro) pode ser associado a muitas classes prêmio, conforme desejado. A mesma classe pode ser designada a diferentes tipos de objetos.

- 30 - As classes prêmio podem ser agrupadas em Pacotes (ou Seguros) por razões de comercialização e vendas (por exemplo, "Full omnium").

- O código de avaliação é usado para classificar as reservas para

uma reivindicação. Uma classe prêmio pode ter tantos códigos de avaliação quanto requeridos, e um mesmo código de Avaliação pode ser designado a diferentes classes Prêmio.

- Os elementos podem ser unidades de informação que identificam as características dos componentes do produto. Por exemplo, no caso da Pessoa objeto segurado, os elementos podem incluir gênero, idade, profissão, etc. No caso de classe Prêmio para seguro de vida ou seguro de incêndio, a soma segurada, etc. pode ser considerada.

Existem os seguintes tipos de elementos:

- Elementos internos: Eles são comuns a todos os produtos, podem ser modificados pelos usuários da Fábrica de Produto e são transparentes para usuários finais (por exemplo, concessão para reintegrar as apólices daquele produto).

- Elementos essenciais: Eles são comuns a todos os produtos, podem ser modificados pelos usuários da Fábrica de Produto e são exibidos nas telas de administração de Apólice screens, tal como, a GUI mostrada nas figuras 28 e 29, uma vez que o usuário final precisa lançar/confirmar o valor selecionado (por exemplo, frequência de pagamento).

- Elementos adicionais: específicos a cada produto, podem ser modificados pelos usuários da Fábrica de Produto e são exibidos nas telas de Administração de Apólice, tal como, a GUI mostrada na figuras 28 e 29 uma vez que o usuário final precisa lançar/confirmar o valor selecionado (por exemplo, uma placa de veículo segurado)

- O método implementado por computador permite particularmente que uma pessoa não versada na técnica de programação para configurar o código fonte de uma maneira simples e eficiente, de modo que os ditos dois ou mais arquivos executáveis interrelacionados possam ser criados no lado de servidor e possam ser executados no lado de servidor.

- Em particular, os arquivos executáveis, quando executados, proporcionam um meio para criar documentos individualmente ao permitir o ajuste através de uma GUI de todas as variáveis/parâmetros, limites, condições, circunstâncias, elementos necessários, etc.

De acordo com figura 1a, para o produto geral, assim como para cada componente do produto, o criador pode escolher um ou mais elementos, validações, operações e textos/parágrafos. De maneira exemplificativa, o objeto do produto "propriedade de conforto domiciliar" pode ser uma "casa".

5 Adicionalmente, os objetos adicionais podem ser introduzidos, tal como, por exemplo, "apartamento".

Além disso, uma cobertura que é selecionada pode ser "construção". Adicionalmente, "conteúdos" de coberturas, "furto" e "auxílios moradia" são selecionados no exemplo de acordo com a figura 1a. Para a cobertura
10 de "construção", os códigos de avaliação "fogo", "explosão", "inundação", "relâmpago" e "terremoto" e o elemento "ano de construção" são selecionados.

2. Módulos de Fábrica de Produto

A Fábrica de Produto Info2000 compreende quatro módulos básicos, ou seja, definições de produto, validações, regras de classificação e
15 texto/parágrafos. Além dos quatro módulos básicos, módulos adicionais podem ser integrados. Uma visão geral exemplificativa dos módulos e sua relação é mostrada na figura 1b.

Definição de Produto

20 Este módulo permite que o criador do produto configure a definição de produto, conforme descrito acima com relação à figura 1a. Toda a informação que refere-se à definição de produto é armazenada no modelo de dados de Fábrica de Produto, junto com a informação lançada nos próximos módulos.

25 Em outras palavras, o objetivo deste módulo é definir a estrutura de produto e permitir a entrada de dados de produto e seus diferentes elementos constituintes (objetos, classes Prêmio, códigos de Avaliação, e elementos):

- Tipo de objeto das características de múltiplas situações e/ou
- 30 - Classes prêmio designadas a cada tipo de objeto e/ou
- Códigos de avaliação designados a cada tipo de classe prêmio e/ou

- Elementos requeridos para cada componente da estrutura de produto:

Definição de componentes (nome, descrição, tipo e tamanho) ao usar um dicionário comum para todos os produtos e/ou

5 Designação de elementos aos componentes de produto e/ou

Relação com os componentes de produto (valores, visibilidade, capacidade de modificação possíveis e padrões, etc.).

Um ou mais dos seguintes elementos – relacionados à definição de produto - podem ser adicionados:

10 Agrupamentos – todos os componentes de Fábrica de Produto podem ser agrupados (produto, objeto, classe prêmio, código de avaliação, elemento). Um tratamento definido mostra o objetivo de agrupamento (por exemplo, resseguro).

15 Modalidades – criadas para classes prêmio e que permitem a seleção de elementos da classe prêmio original a ser incluída na modalidade. As características de elementos descritos (valor padrão, mínimo, máximo) são especificadas ao estabelecer a relação entre as classe prêmio e a modalidade.

20 Opções – A definição de opções tem um duplo propósito: facilitar a subscrição e estabelecer as modalidades/classes prêmio predefinidas. As opções são criadas no nível de objeto.

Validações

25 Este módulo permite que o criador do produto configure as regras a serem executadas e validadas durante o ciclo de vida do produto na administração de apólice, isto é, quando executa dois ou mais arquivos executáveis interrelacionados a fim de produzir, por exemplo, um documento de seguro. Uma possível validação pode ser definida, por exemplo, para validar se um motorista é maior de 18 anos, quando executa dois ou mais arquivos executáveis, em que uma variável é a idade de um motorista. A Fábrica de

30 Produto permite a definição de variáveis/parâmetros, tal como a idade, quando define uma validação que será realizada durante o tempo de execução dos dois ou mais arquivos executáveis interrelacionados (isto é, durante

o tempo de execução do produto). Estas validações, como quaisquer outros componentes de produto são reutilizáveis entre os produtos. Em outras palavras, a validação serve para verificar se a idade de um motorista, como uma variável exemplificativa, usada quando define, por exemplo, que um objeto do produto é maior que um dado valor limítrofe. A dita validação também pode ser usada, quando define um objeto diferente do produto e/ou a cobertura e/ou o próprio produto.

Em outras palavras, o objetivo deste módulo é permitir a introdução de validações necessárias associadas a um produto (técnicas ou comerciais).

Uma validação é composta por uma ou algumas condições lógicas, unidas por meio de operadores Booleanos, onde cada condição lógica é feita de operando(s) e um operador de lógica. O resultado da validação é VERDADEIRO ou FALSO.

Um ou mais dos seguintes componentes podem ser encontrados em uma validação:

Nome e descrição de uma ou mais validações.

Operandos, tais como, valores elemento, resultados de operação, acesso às tabelas de corporação, etc.

Os operadores de comparação (maior que, menor que, igual), compatibilidade, compulsórios, etc.

Conectores lógicos para validações que podem compreender E e/ou OU.

Sentido de validação que pode compreender Verdadeiro e/ou Falso.

Similares aos elementos, as validações devem ser associadas ao produto, objeto, classe prêmio, etc., uma vez que elas são definidas. Uma vez que uma validação é definida, a mesma pode ser reutilizada em outros componentes de um produto ou em diferentes produtos.

Um exemplo de uma validação é mostrado na captura de tela incluída na figura 1c, que mostra uma GUI exemplificativa. Neste exemplo (no pseudocódigo):

SE

o Comercio for Escavação (10030)

OU

Revestimento de Estrada (10028)

5

E

As atividade incluem bombeamento/despejo de concreto

ENTÃO ... (resultado de validação)

Consequentemente, usando o método de acordo com este pedido, um criador do produto pode desenvolver isto sem uma única linha de código ser escrita. Deste modo, de maneira vantajosa, o projeto técnico, codificação e/ou teste de unidade pode ser eliminado do ciclo de vida de desenvolvimento de produto.

A figura 2 mostra uma visão geral sobre as possíveis configurações das validações. Em particular, a figura 2 mostra a possível configuração de cada validação, que usa condições lógicas N, onde N pode ser um número natural, em particular, 2, 3, 4, 5, 6, 8, 10, 15, 20, etc. Além disso, mostra-se de maneira exemplificativa na figura 2 os possíveis componentes de cada condição. Cada uma das condições lógicas N pode compreender operandos M, onde M pode ser um número natural, em particular, 2, 3, 4, 5, 6, 8, 10, 15, 20, etc. Além disso, para cada condição lógica, os M operandos lógicos podem ser conectados através de um dos operadores de lógica. Consequentemente, proporciona-se para cada condição lógica operadores de lógica M-1. Os operandos lógicos podem compreender um ou mais: Elementos, Acessos, Operações, funções predefinidas, tipo de objeto, Classe prêmio, constante Numérica, constante Alfanumérica, Rotina, etc. O operador de lógica pode compreender um ou mais: EQ=igual, NE=não igual, GT=maior que, GE=maior que ou igual a, LT=menor que, LE=menor ou igual a, TC=cobertura contraída, TNC=cobertura não contraída, TI= cobertura incompatível, TD=cobertura que depende de, etc.

As condições lógicas N podem ser conectadas a um operador Booleano, tal como "E", "OU", "(", ")", etc.

Também na figura 2, mostra-se um exemplo adicional de uma

possível validação, ou seja, para verificar, se a idade de uma pessoa a quem os documentos de seguro são criados tem uma idade superior a 100 anos. Se a dita validação descobrir que a idade, como um parâmetro preferido, for maior que 100 anos, uma ação específica é realizada, por exemplo, cobranças mensais de 7,50 EUROS serão cobradas.

O módulo "Validações" pode ser acessado quando usar o sistema de acordo com o presente pedido, seguindo ao caminho *Sistema>Prod. Conf.>Validações*. A partir daqui, novas validações podem ser adicionadas, modificadas ou apagadas.

A figura 3 mostra a captura de tela de uma interface gráfica de usuário exemplificativa (GUI) do sistema baseado em computador de acordo com o presente pedido, quando adiciona ou modifica uma validação, que foi criada para propósitos de ilustração. Em outras palavras, a captura de tela, de acordo com figura 3, mostra uma validação que estabelece basicamente que a validação deve ocorrer se a soma dos conteúdos exceder a figura em 2.500.000. A parte superior desta captura de tela pode conter o nome, descrição, data e operadores. O retângulo realçado é o lugar em que a validação real é exibida. Este módulo será usado para validar os dados de apólice como um acionador de um cálculo dentro de uma operação ou como um acionador de parágrafos. Se esta validação não ocorrer, então, a mesma irá lançar uma mensagem de erro. Em outras palavras, a figura 3 mostra uma GUI 1 que exibe texto claro, em particular, que tem, por exemplo, campos 2a, 2b, 4a, 4b, etc., para lançar texto, em que o usuário pode selecionar operadores específicos. Esta seleção através da GUI é associada ao código legível por máquina na linguagem de máquina. Além disso, o usuário pode especificar variáveis e/ou parâmetros que são submetidos, por exemplo, aos operadores. Através da GUI, as variáveis e/ou parâmetros são ajustados na linguagem legível por máquina. Na figura 3, uma condição é exibida em texto claro. Em particular, os campos 2a, 2b, são usados para lançar um período de tempo durante o qual a validação é aplicável. Como um exemplo, a data de início lançada através do campo 2a é escolhida na figura 3 como 01 de janeiro de 1900 e a data final lançada através do campo 2b é 31 de dezembro de

9999. Durante este tempo, a validação será realizada, quando dois ou mais arquivos executáveis interrelacionados forem executados.

O nome e uma curta descrição da validação são lançados usando os campos 4a, 4b, em que os ditos campos são usados para descrever brevemente a validação e, deste modo, permitir uma simples identificação da validação, por exemplo, na tabela de definição, conforme mostrado na figura 4 (vide abaixo). Adicionalmente, a GUI 1 de acordo com a figura 3 contém os campos 6a, 6b para lançar os operandos que definem a validação, assim como, o operador no campo 8. A validação, assim, configurada e definida é mostrada na GUI 1, na janela 10, em que a linha que define a validação foi realçada por uma caixa 12 (não mostrada, quando exibe a GUI) para uma fácil visibilidade. A GUI 1 compreende a definição uma operação que é chamada de OP_DEFAULT_8 e usa os CONTEÚDOS de SOMA de validação 4 > LIM como um acionador de cálculo 01 e 05 que significa que o cálculo será realizado dependendo do resultado desta validação. Então, no cálculo, dependendo do resultado, C1 ou C2 podem ser acionados. C3 é sempre acionado. Além disso, a descrição acima mantém, onde aplicável, a similaridade das GUIs, conforme mostrado nas outras figuras.

Como um exemplo, todas as validações são armazenadas na tabela KTPM34T. Em outras palavras, a tabela KTPM34T é uma tabela de definições exemplificativa. A figura 4 mostra uma captura de tela que compreende a definição da validação, conforme selecionada com a GUI de acordo com a figura 3. As linhas realçadas com uma caixa (não mostrada quando exibe a GUI) são as que contêm o código exclusivo da validação, neste exemplo, "0000006619", datas válidas, neste exemplo, de "19000101" a "99991231" que foram lançadas através dos campos 12a, 12b da GUI 1 e da definição da validação. Também, nas figuras a seguir, as caixas usadas para realçar normalmente não são exibidas pela GUI do sistema computadorizado.

A validação será executada apenas se a soma dos conteúdos for > 2.500.000. Estes valores sem os quais a validação não irá disparar são armazenados na tabela KTPA78T. Em outras palavras, a tabela KTPA78T é

uma tabela de condições exemplificativas. As figuras 5a e 5b, que são duas capturas de tela, realçam a condição definida com a GUI 1 de acordo com a figura 3 e também com os ditos valores anteriormente mencionados. As capturas de tela mostram duas linhas separadas como R1 e R2 que não podem se adequar apenas a uma captura de tela, de modo que elas sejam exibidas nas figuras 5a e 5b.

Portanto, os conteúdos são lidos a partir da tabela anterior KTPA78T, um excerto desta é mostrado nas figuras 5a e 5b, de modo que se a soma for maior que 2.500.000, então, o programa é executado. Isto é que representa o conteúdo da validação.

Em outras palavras, através da GUI 1 mostrada na figura 3, as definições da validação são criadas em uma ou mais tabelas, no presente exemplo, entradas são criadas nas tabelas mostradas nas figuras 4, 5a e 5b, em que em uma tabela de definição (figura 4), a definição da validação, em geral, é proporcionada e em outra tabela, isto é, na tabela de condição (figuras 5a e 5b) a própria condição e os valores de variáveis/parâmetros usados pela validação são definidos.

Um gerador sob encomenda (fonte COBOL KTPLP007), a fim de gerar automaticamente o código em copybooks irá acessar esta tabela KTPA78T, conforme mostrado nas figuras 5a, 5b e realizar as seguintes instruções representadas em pseudocódigos. A seguinte lógica em pseudocódigos representa o meio no qual o programa opera e se converte em fonte COBOL que é configurada nas tabelas mencionadas:

Etapa 1: Ler o primeiro registro a partir de KTPA78T (R1) e

Etapa 2: Realizar

Etapa 2.1: Escrever nova linha com a informação de KTPA78T

Etapa 2.2: Ler o próximo registro a partir de KTPA78T

Até o final de KTPA78T.

Etapa 3: Escrever novas "Linhas Finais".

O processo de acordo com as subetapas 2.1 e 2.2 é iterativo/repetido, isto é, o dito processo termina quando o mesmo alcança o fim das fileiras. No presente exemplo, de acordo com as figuras 3, 4, 5a e 5b

duas iterações serão realizadas.

Referindo-se à natureza matemática das regras inseridas em KTPA78T, os ditos componentes tabela são definidos na validação e/ou operação.

- 5 De maneira exemplificativa, conforme mostrado nas figuras 5a e 5b, uma nova regra codificada por 000000619 que é criada consiste em uma LÓGICA CONDICIONAL (portanto, uma validação), através de valor "CL" no campo TCTCP78T que é o primeiro operando (campo CDORD78T com valor "01") e o operando 0000006178 (campo CDCCH78T que é uma função pre-
- 10 definida) deve ser maior (campo TCOBH78T com valor ">") que o segundo operando (segunda fileira na figura 5a) que é numérico (campo TCTCH78T com valor N) e seu valor é 2.500.000.

- Então, aqui, a natureza matemática desta regra codificada é o campo TCOBH78T com o valor ">", onde o código legível por máquina define
- 15 que o valor derivados de variáveis 0000006178 deve ser comparado com um valor fixo, isto é, 2.500.000.

Possíveis indicadores podem ser ">", "<", "=>", "=<", "=", etc.

- O gerador feito sob encomenda é projetado e adaptado para reconhecer condições específicas. Como um exemplo, a tabela KTPA78T
- 20 compreende no campo TCTCP78T, o valor "CL" que significa Lógica de Condição e é igual à operação "IF". Em outras palavras, de maneira exemplificativa, na(s) tabela(s) um ou mais parâmetros que podem ser armazenados compreendem informação sobre condições lógicas. O gerador feito sob encomenda é adaptado para ler o(s) parâmetro(s), identificar a lógica do(s) pa-
- 25 râmetro(s) e usar o(s) parâmetro(s) para criar código fonte cobol (ou código fonte em qualquer outra linguagem legível por máquina, tal como, FORTRAN, C, C++, PASCAL, JAVA, etc.

- Como um resultado, a validação é gerada em linguagem legível por computador, por exemplo, em código fonte COBOL. A captura de tela
- 30 mostrada na figura 6 contém o código fonte do programa resultante, que pode ser usado em uma etapa posterior, quando gerar o arquivo executável. Em outras palavras, a tabela mostrada na figura 6 é uma tabela de código

exemplificativo, em que o código fonte é equivalente à condição, à medida que lançado através da GUI mostrada na figura 3.

Portanto, usando o sistema baseado em computador, de acordo com o presente pedido, o criador de um produto, isto é, dois ou mais arquivos executáveis interrelacionados, pode definir uma ou mais validações em texto claro usando a GUI 1, conforme mostrado na figura 3. O conteúdo da GUI mostrado na figura 3 é transferido para uma tabela de definição, conforme mostrado na figura 4. Os valores numéricos e/ou condições matemáticas da validação são armazenados em uma tabela de condição, conforme mostrado nas figuras 5a e 5b. Usando as condições matemáticas e os valores numéricos, à medida que armazenados na tabela de condição, de acordo com as figuras 5a e 5b, o código fonte é automaticamente gerado. O código fonte é armazenado em uma tabela de código, conforme mostrado na figura 6. É possível que as condições sejam geralmente armazenadas em uma ou mais tabelas de condição e que os valores numéricos, por exemplo, de variáveis/parâmetros usados nas condições armazenadas na(s) tabela(s) de condição, sejam armazenados em uma ou mais tabela(s) de valor e/ou uma ou mais tabelas de variável/parâmetro.

- Acessos

Este componente de Fábrica de Produto, isto é, o sistema/método baseado em computador, de acordo com o presente pedido para criar dois ou mais arquivos executáveis interrelacionados, que é parte de um módulo, permite através do campos chave para a busca de dados em uma tabela de corporação. Os acessos também são reutilizáveis entre os produtos, isto é, diferentes métodos/sistemas baseados em computador que compreendem dois ou mais arquivos executáveis interrelacionados.

É possível adicionar, modificar ou apagar um Acesso ao seguir o caminho *Sistema>Prod. Conf.>Acessos*. As capturas de tela, conforme mostrado, nas figuras 7a, 7b, 8a, 8b e 9 contêm o exemplo de um Acesso a uma tabela com todas os seus resultados chave e econômicos. Cada Acesso tem um conjunto de chave(s) e o(s) resultado(s) associado(s). A seguir, encontra-se um exemplo de um Acesso "ACC_SIC_2ND". As chaves são realçadas

nas duas capturas de tela, de acordo com as figuras 7a e 7b, à medida que existem muitas para se ajustar em uma única captura de tela.

Os resultados econômicos mencionados acima são realçados nas duas capturas de tela, de acordo com as figuras 8a e 8b.

5 Os Acessos são armazenados na tabela KTPM38T. A captura de tela, de acordo com a figura 9, contém a definição do exemplo acima em uma tabela de definição. Os elementos relevantes estão realçados, incluindo seu código exclusivo "0000001280".

Os resultados econômicos deste Acesso, tais como, os dados
10 econômicos relevantes que podem ser considerados quando produzem o código fonte e, posteriormente, nos arquivos executáveis, são armazenados na tabela KTPA30T. Estes são os mesmos que aparecem no exemplo do sistema/método implementado por computador, conforme mostrado nas figuras. Eles são realçados nas capturas de tela, conforme mostrado nas figuras
15 10a e 10b, que representam uma tabela de variável/parâmetro.

A Tabela KTPA37T com as chaves para este Acesso particular é exibida nas duas capturas de tela, de acordo com as figuras 11a e 11b, que mostram uma tabela de condição.

O código do programa é exibido nas quatro capturas de tela, de
20 acordo com as figuras 12a a 12d, que mostram uma tabela de código. Os resultados e as chaves são selecionados e, então, o Acesso é executado. O resultado é a geração de um Acesso.

O gerador (fonte COBOL KTPLP007), a fim de gerar automaticamente o código e nos copybooks irão acessar ambas as tabelas
25 (KTAP30T e KTPA37T) e realizar as seguintes instruções representadas no pseudocódigo. As seguintes instruções no pseudocódigo representam como o programa opera e se converte em fonte COBOL que é configurada nas tabelas mencionadas:

Etapa 1

30 Etapa 1.1 - Ler o primeiro registro a partir de KTPA30T (R1) e
Etapa 1.2 - Realizar
Etapa 1.2.1 - Escrever nova linha com a informação de KTPA30T

(L1)

Etapa 1.2.2-. Ler o próximo registro a partir de KTPA30T (R2)
Até o final de KTPA30T.

Etapa 1.3 - Escrever novas "Linhas Fixas"

5

Etapa 2

Etapa 2.4 - Ler o primeiro registro a partir de KTPA37T (R1) e

Etapa 2.5 - Realizar

Etapa 2.5.1 - Escrever nova linha com a informação de KTPA37T

(L1)

10

Etapa 2.5.2 - Ler o próximo registro a partir de KTPA37T (R2)
Até o final de KTPA37T.

Etapa 2.6 - Escrever novas "Linhas Fixas"

Etapa 3

Etapa 3.7 - Ler o primeiro registro a partir de KTPA30T (R1) e

15

Etapa 3.8 - Realizar

Etapa 3.8.1 - Escrever nova linha com a informação de KTPA30T

(L1)

Etapa 3.8.2 - Ler o próximo registro a partir de KTPA30T (R2)
Até o final de KTPA30T.

20

Etapa 3.8 - Escrever novo erro de manipulação de "Linhas Fixas", etc...

No pseudocódigo acima, as "linhas fixas" usam a informação do código de tabela e código de acesso KTPM38T.

25 As subetapas 1.2.1 e 1.2.2 podem ser iterativas/repetidas. Também, as subetapas 2.5.1 e 2.5.2 podem ser iterativas/repetidas. Também, as subetapas 3.8.1 e 3.8.2 podem ser iterativas/repetidas.

O pseudocódigo acima é refletido no código, conforme mostrado nas figuras 12a a 12d, em que a figura 12a representa a etapa 1, a figura 12b representa a etapa 2 e as figuras 12c e 12d representam a etapa 3, que
30 já começa na linha 061946 na figura 12b.

- Regras de Classificação

Através do módulo "regras de classificação", o criador do produto

define as operações (feitas de um ou mais cálculos) usadas para calcular a taxa de produto (os prêmios foram calculados no nível de cobertura). Não apenas os elementos podem ser usados nos cálculos, porém, os valores (por exemplo, fatores) armazenados nas tabelas de corporação podem ser usados também através dos "Acessos". Os programas de classificação externa também podem ser "capturados" nestes cálculos. Uma visão geral esquemática é mostrada na figura 13, que é similar à figura 2. Em particular, a figura 13 mostra uma operação que pode compreender um grupo de cálculos, tais como, cálculos n , em que n é um número natural. Em particular, n é igual a 1, 2, 3, 4, 5, 6, 8, 10, 15, 20, etc. Um cálculo pode ser uma validação prévia que compreende operandos e operadores, conforme descrito com referência à figura 3.

As capturas de tela de um exemplo gráfico são mostradas nas figuras 14a a 14d, que podem ser criadas pelo exemplo do sistema/método implementado por computador, conforme mostrado pelas figuras, quando seguem o caminho *Sistema>Prod. Conf.>Cálculos*. A partir daqui, as operações podem ser adicionadas, modificadas ou apagadas. Por exemplo, as figuras 14a a 14c representam um composto de cálculos para uma operação chamada de "OP_DEFAULT_8". Cada linha é separadamente realçada em uma respectiva figura. A primeira linha (na figura 14a) contém o cálculo "C1". A segunda linha (na figura 14b) contém o próximo cálculo, "C2" e a última linha (na figura 14c) contém o último cálculo "C3".

Portanto; o módulo de classificação permite a realização de todas as operações financeiras necessárias para a criação de um produto técnico ou comercial, assim como o acesso às tabelas necessárias para realizar qualquer uma das ditas operações.

Um acesso permite a consulta dos valores armazenados em uma tabela e a configuração das condições para a consulta particular. Pode-se especificar o seguinte:

30 Chave(s), que pode(m) compreender o(s) campo(s) da tabela mediante a qual uma condição de acesso é estabelecida.

Operador(s), que pode(m) compreender igual, maior que, menor

ou igual a, etc.

Operando(s) que pode(m) compreender constante(s) através das quais a chave é comparada.

Campo(s) de tabela que correspondem ao resultado do acesso.

5 Um ou mais dos seguintes componentes podem ser encontrados em uma operação:

Nome e descrição de uma operação.

Cálculos que contêm

- operandos que podem ser constantes numéricas, valores numéricos de elementos, resultados de outros cálculos ou operações, resultados numéricos de acesso às tabelas, rotinas, etc. e/ou

- operadores: +, -, *, / e/ou

- circuitos e funções de soma e/ou

- funções predefinidas que podem compreender grupos de cálculo

15 lo padronizado que fornecem um resultado numérico de acordo com os parâmetros de uma entrada.

Também é possível que pelo menos uma validação possa ser adicionada antes de uma operação. As operações podem ser designadas a uma classe prêmio e/ou um objeto e/ou a conceitos econômicos específicos.

20 Uma vez que uma operação é registrada, a mesma pode ser reutilizada em outros componentes de um produto ou em produtos diferentes.

Outro exemplo de classificação é mostrado na captura de tela incluída na figura 14d, que mostra uma GUI exemplificativa. Neste exemplo (em pseudocódigo):

25 Taxa de Prêmio de Seguro =

Taxa Antes do Prêmio *

Alíquota de Imposto de Prêmio de Seguro / 100

A GUI exemplificativa permite o tipo direto da operação no campo especificado, usando a sintaxe correta. A dita operação pode ser validada usando o botão '...'. Também é possível procurar o operando e extrair o operando a ser usado ao usar o botão "Operando". Portanto, assegura-se que uma operação existente seja usada.

No conteúdo dos cálculos acima 3 níveis foram definidos:

- C1 (referido como linha 01 na figura 14a) é cálculo baseado em uma validação que é executada se a validação for FALSA, isto é, o cálculo C1 é realizado se o resultado da validação "CONTEÚDOS DE SOMA 4 > LIM" for "falso". A validação "CONTEÚDOS DE SOMA 4 > LIM" foi definida de maneira exemplificativa pela figura 3 acima.

- C2 (referido como linha 05 na figura 14b) é um cálculo baseado na validação "CONTEÚDOS DE SOMA 4 > LIM". O cálculo C2 é executado se a dita validação for "verdadeira".

10 - C3 (referido como linha 10 na figura 14c) é um cálculo que sempre é executado.

A operação real é definida para o criador do produto e armazenada na tabela KTPM56T, isto é, na tabela de definição. Esta tabela é aquela em que todas as operações são armazenadas. A Tabela KTPM56T é mostrada na captura de tela de acordo com a figura 15.

15 As definições deste cálculo são contidas na tabela KTPA57T, como uma tabela de condição exemplificativa, que é mostrada de maneira exemplificativa nas duas capturas de tela, de acordo com as figuras 16a e 16b. Para cada linha (R1, R2, R3), que é armazenada na tabela KTPA57T (figuras 16a e 16b), o formato do cálculo e quaisquer variáveis associadas são definidos.

O conteúdo do cálculo é armazenado na tabela KTPA78T, como uma tabela de variável/parâmetro exemplificativa, que é mostrada de maneira exemplificativa nas duas capturas de tela, de acordo com as figuras 17a e 17b. De maneira resumida, a tabela KTPA78T mantém as relações matemáticas entre os operandos e os operadores, então, no final a dita tabela armazena como os elementos ou componentes são relacionados em uma regra matemática.

30 O código do programa é mostrado de maneira exemplificativa na tabela, de acordo com as capturas de tela na figuras 18a a 18c, que é uma tabela de código exemplificativa. A partir das seguintes instruções no pseudocódigo, pode-se ver o acontece em cada linha. Como nas validações e

acessos, o gerador (fonte COBOL KTPLP007) gera um código automaticamente nos copiadores que irão, por sua vez, acessar as tabelas (KTPA57T e KTPA78T) e irão, então, informar estas instruções de pseudocódigo. Desta maneira o programa converte o conteúdo de uma tabela em fonte cobol (em pseudocódigo):

Etapas 1:

1.1 - Ler o primeiro registro a partir de KTPA57T e escrever

1.2 - Realizar

1.2.1 – Se existe uma validação associada ao cálculo:

10 1.2.1.1 – Escrever a sentença REALIZAÇÃO para executar a validação.

1.2.1.2 – Escrever o SE para avaliar o resultado de validação (Verdadeiro ou Falso)

1.2.2 - Escrever REALIZAÇÃO para executar o cálculo. (C)

15 1.2.3 - Se existe uma validação associada ao cálculo:

2.3.1 – Escrever Final de linhas e sair

1.2.2 - Ler o próximo registro a partir de KTPA57T

Até o final de KTPA57T.

20 Como um resultado do circuito acima, o gerador gera o formato do cálculo baseado no que é lido a partir de KTPA57T, e o código é gerado no copiador TP3CCYAU, à medida que o mesmo é mostrado na captura de tela, de acordo com a figura 18a.

25 O cálculo de nível 01 (vide definição dos níveis acima) é realizado quando a validação é "falsa", então, o segundo nível 05 é realizado quando a validação é "verdadeira" e o terceiro nível 10 será executado não submetido a qualquer validação. Os níveis são indicados na figura 18a pelas respectivas caixas rotulada "F" (cálculo de nível 01), "T" (cálculo de nível 05) e "Cálculo" (cálculo de nível 10).

30 Finalmente, o conteúdo dos 3 níveis do cálculo (C1, C2 e C3) ou o nível 01, 05 e 10 é escrito no mesmo copiador TP3CCYAU. O pseudocódigo ou as instruções que os programas seguem, a fim de escrever os conteúdos são os seguintes (pseudocódigo para acesso de tabela KTPA78T):

Etapa 2:

2.1 - Ler o primeiro registro a partir de KTPA78T e

2.2 - Realizar

2.2.1 - Escrever nova linha com a informação de KTPA78T

5 2.2.2 - Ler o próximo registro a partir de KTPA78T

Até o final de KTPA78T.

Subetapas 2.2.1 e 2.2.2 podem ser iterativas/repetidas.

Como um resultado deste circuito (etapas 2.2.1 e 2.2.2) neste copiador mencionado acima o CONTEÚDO destes cálculos (C1, C2 e C3 ou
10 os níveis 01, 05 e 10) também é escrito. Na captura de tela, de acordo com as figuras 18b e 18c, mostra-se como os cálculos são realizados em cada nível. Isto significa que a REALIZAÇÃO (instrução Cobol que executa o que é definido no interior) que é chamada na ETAPA 1 é definida neste ponto na ETAPA 2, conforme mostrado nas figuras 18b e 18c.

15 Textos/Parágrafos

O módulo "textos/parágrafos" permite a definição dos textos (modelos de documento) a serem gerados para cada produto, isto é, para cada conjunto de dois ou mais arquivos executáveis interrelacionados, assim como as regras de disparo para os parágrafos (por exemplo, se a cobertura
20 A for selecionada, a cláusula XXX será impressa). Uma visão geral esquemática é mostrada na figura 19 (que é similar às figuras 2 e 13). Na figura 19, a criação de acordo com os textos/parágrafos módulo é mostrada. Como um exemplo, o "texto" pode compreender inúmeros parágrafos, tais como, 1, 2, 3, 4, 5, 6, 10, 15, 20, etc. parágrafos, cada um destes pode ser individual-
25 mente direcionado e individualmente associado, por exemplo, a pelo menos um produto e/ou pelo menos um objeto segurado e/ou pelo menos uma cobertura, etc. Na figura 19, o "Objeto Segurado" é associado ao "Parágrafo 1". Também é possível que 2 ou mais parágrafos sejam associados ao dito "Objeto Segurado". Além disso, a "Cobertura 1" é associada a um parágrafo, ou seja, "Parágrafo 2". "Cobertura 2" é associada a um parágrafo, ou seja, "Pa-
30 rágrafo 3". É possível que cada cobertura seja associada a dois ou mais parágrafos e/ou que cada cobertura seja associada a um parágrafo idêntico.

Como um exemplo, a "Cobertura 1" pode ser associada ao "Parágrafo 1" e a "Cobertura 2" pode ser associada ao "Parágrafo 2" e o "Parágrafo 3". Também é possível, neste exemplo, que a "Cobertura 1" seja associada ao "Parágrafo 1" e ao "Parágrafo 2".

5 Como nas validações, acessos e operações, as tabelas são usadas para definir e armazenar variáveis. A Tabela KTPA65T é a base para os documentos gerados para um produto.

Os parágrafos têm um código fixo que significa que naquela tabelas existem parágrafos que são associados a cada código de cobertura e, também, a cada produto, isto é, para cada conjunto de dois ou mais arquivos executáveis interrelacionados.

Para disparar os parágrafos também existem validações que são associadas aos mesmos, isto significa que um parágrafo de texto pode ser disparado com base no fato de que a cobertura que é associada é contratada e, também, se a mesma tem uma validação associada a isto. Portanto, este módulo permite a designação/não designação de cláusulas específicas e textos necessários na definição de produto. Isto permite a designação de texto(s) aos parágrafos correspondentes em qualquer nível de definição de produto. Isto permite mostrar se um parágrafo é opcional ou compulsório e, também, associar condições lógicas aos parágrafos.

As capturas de tela, de acordo com as figuras 20a a 20c representam um exemplo desta definição KTPA65T de tabela.

As capturas de tela, de acordo com as figuras 21a e 21b contêm a parte do programa COBOL que acessa os campos da tabela KTPA65T e lê os valores que documentação é projetada para ser gerada. Como um resultado de acesso à tabela KTPA65T, o programa pode gerar a documentação requerida.

Aqui, o processo não é realmente o mesmo que os outros níveis explicados como o executável que é chamado a fim de disparar os parágrafos (e obviamente os documentos), ler os registros associados a um parágrafo particular e mover toda a informação armazenada em uma variável interna através de um Instrução Cobol chamada CURSOR, conforme mostrado nas

capturas de tela, de acordo com as figuras 21a e 21b.

Finalmente, após a leitura da informação armazenada para cada parágrafo, no mesmo copiador (executável), TPSTCDBD é verificada se um certo parágrafo precisa ser disparado ou não apesar do fato de ser contratado. Isto também é verificado através do código automatizado gerado com base nas validações associadas aos parágrafos que também foram descarregados naquele copiador, conforme mostrado na captura de tela de acordo com figura 22.

Além do que foi dito acima, um ou mais dos módulos de suporte podem ser incluídos:

- Simulação (Simulador)

O objetivo deste módulo é permitir que o usuário verifique as operações e/ou cálculos associado à definição de produto e use os dados de portfólio para simular os resultados de alterações na definição de produto (por exemplo, simular o impacto no prêmio em dinheiro devido a uma alteração no preço de produto).

Coerência

O objetivo principal deste módulo é assegurar que a definição de produto realizada a partir de módulos de definição, validação e taxa/classificação oferece integridade suficiente

para permitir a geração de produto (objeto executável) e/ou que a Administração de apólice e pedidos de Reivindicações possam realizar as operações com base no produto definido.

- Utilidades

Este módulo proporciona uma ou mais das seguintes facilidades: Cópia de produtos.

Introdução e uso de tabelas biométricas (por exemplo, tabelas de mortalidade).

Criação de variáveis de vida (símbolos de comutação).

Geração automática de descrição nas tabelas de corporação.

Além do que foi dito acima, um módulo de gerenciamento de competência pode ser incluído.

- Gerenciamento de Competência

Os parâmetros de competência para elementos e/ou para validações podem ser definidos a partir da aplicação de Fábrica de Produto. Ao definir um elemento e/ou validação como um parâmetro de competência, seu comportamento pode ser modificado, de acordo com papel e/ou localização geográfica e/ou acesso de usuário do elemento na administração de apólice, em que os níveis de autorização podem ser levados em consideração. Este módulo pode ser conectado à aplicação de componentes comum, que basicamente gerencia as competências.

Adicionalmente, um módulo de subscrição pode ser proporcionado.

- Subscrição

Este módulo pode permitir adaptações rápidas de produtos comerciais em nível local e permitir o fechamento de detalhes de negociação e alterar as características de um produto comercial de acordo com os parâmetros predeterminados. Este módulo pode se aplicar ao cliente/grupo de afinidade/segmento alvo/intermediário.

3. Produto Gerador

Após configurar o produto usando os módulos anteriores, isto é, após a composição de uma combinação de códigos legíveis por máquina na linguagem de máquina ao selecionar e associar duas ou mais seções de código usando a GUI, a Fábrica de Produto Gerador transforma os ditos dados, que podem ser configurados a partir das tabelas, nos arquivos executáveis COBOL que contém a base de um produto, isto é, todas as informações e regras de produto (exceto para aqueles valores mantidos nas tabelas de corporação). No contexto do presente pedido, o termo "programa executável" pode ser sinônimo para o termo "arquivo executável" ou o termo "programa executável" pode compreender dois ou mais arquivos executáveis interrelacionados. Através dos módulos de Administração de Apólice, apólices, validações de dados, etc. podem ser gerados para o produto.

Este Produto Executável é chamado a partir dos serviços de Administração de Apólice através de interfaces padrão, conforme mostrado

de maneira exemplificativa na figura 23. De acordo com a figura 23, existe um serviço de Administração de Apólice que é o KCILMZ06 que pode chamar as executáveis quando o usuário ou quando a ação é VALIDAR ou CLASSIFICAR.

- 5 Ambas as ações podem ser disparadas em três diferentes níveis:
 Apólice
 Objeto
 Cobertura.

 Agora, o serviço de Administração de Apólice é sempre o mesmo
 10 (KCILMZ06), porém, as executáveis a partir da Fábrica de Produto que realizam a ação de VALIDAÇÃO OU CLASSIFICAÇÃO são dinâmicas. Isto significa que o serviço KCILMZ06 está acessando uma ou mais tabelas, onde o nome das executáveis que precisam usar em cada nível e em cada ação é armazenado. Esta tabela é, por exemplo, KTPM01T.

15 4. Resumo de Fluxos de Processo de Fábrica de Produto

 A figura 24 mostra um fluxograma que resume os fluxos de processo principais descritos ao longo de todo o documento. A figura 24 descreve a funcionalidade da "Fábrica de Produto", ou seja, para gerar particularmente programas de alta complexidade a partir de uma entrada de dados
 20 inicial.

 As etapas S1 a S3 são realizadas, de preferência, no lado de cliente. As etapas S4 a S6 são realizadas, de preferência, no lado de servidor.

 Na etapa S1, o dado relevante para definir os dois ou mais arquivos executáveis interrelacionados é lançado por um usuário do sistema/método baseado em computador, de acordo com o presente pedido. O sistema/método baseado em computador também pode ser referido como "Fábrica de Produto". Em particular, o dado é lançado usando uma GUI, conforme mostrado de maneira exemplificativa nas figuras, tal como a figura 3.

30 Na etapa S2, o dado lançado é armazenado em uma ou mais tabelas, conforme também mostrado nas respectivas figuras, tais como, figuras 4, 5a e 5b. As etapas 1 e 2 podem ser realizadas de uma maneira repeti-

da, em que o dado pode ser subsequentemente lançado (etapa 1) e o dito dado pode ser armazenado em uma ou mais tabelas (etapa 2).

Na etapa S3, a verificação implementada por computador da coerência é realizada, em que uma ferramenta de coerência pode ser usada. A ferramenta de coerência pode compreender uma GUI para comunicação com o usuário do sistema. Em particular, a ferramenta de coerência pode ser adaptada para produzir um resultado do teste de coerência no texto claro. De maneira vantajosa, a coerência do código na linguagem legível por máquina pode ser verificada em sua totalidade no lado de cliente, particularmente, sem compilação. Os dois ou mais arquivos executáveis podem, então, ser criados no lado de servidor (vide etapas S4 a S6) e executarem, com risco minimizado de erros de tempo de execução.

Na etapa S4, os processos que compreende, por exemplo, validações, acessos, etc. podem ser gerados e, em particular, os dois ou mais arquivos executáveis interrelacionados são compilados usando um compilador e, deste modo,

na etapa S5, os dois ou mais arquivos executáveis interrelacionados (referidos, por exemplo, como programas de resultado) são gerados.

Na etapa S6, os dois ou mais arquivos executáveis interrelacionados são testados ao executar os mesmos o lado de servidor.

A seguir, as vantagens do método/sistema implementado por computador, de acordo com o presente pedido, são descritas:

5.1. Evitar problemas de conexão

Esta vantagem será explicada por meio de um exemplo, onde a operação 1 foi projetada da seguinte maneira:

Operação 1 = Elemento 1 + Elemento 2

Em um banco de dados específico, a Operação 1 pode ser conectada ao Produto Técnico (isto é, a cada conjunto de dois ou mais arquivos executáveis interrelacionados) que a mesma corresponde, a fim de indicar para sistema onde o mesmo pode recuperar os valores para os componentes da operação (elemento 1 e elemento 2 neste exemplo). A conexão da operação significa que primeiro os operandos devem ser conectados, neste

caso, o Elemento 1 e o Elemento 2, onde estes elementos devem ser designados ao nível desejado, que pode ser o produto, objeto ou cobertura (os níveis são indicados na figura 1a). Então, uma vez que os elementos são conectados, o sistema sabe onde os valores relacionados ao mesmo são armazenados. O mesmo ocorre com a operação. Como um resultado desta operação, o fato de ser conectado significa que o sistema irá saber onde os valores relacionados ao mesmo são armazenados.

Quando gerar o produto, isto é, os dois ou mais arquivos executáveis interrelacionados, o gerador irá verificar se todas as conexões foram eficientemente estabelecidas. Se não, o processo falha, e uma descrição de erro será escrita no registro de problema.

A fim de evitar falha, a operação pode ser adequadamente conectada antes da geração. Além disso, de acordo com o presente pedido, a ferramenta de coerência é usada antes da geração, a fim de descobrir se as conexões foram eficientemente efetuadas. No caso de uma conexão incorreta ter sido efetuada, a ferramenta de coerência proporciona uma mensagem de erro, que aponta a inconsistência e/ou uma conexão incorreta. A ferramenta de coerência é a capacidade que permite a verificação (antes da geração) da coerência da Fábrica de Componentes de produto definição e suas relações. Então, esta ferramenta valida não apenas a definição, mas também se a relação construída também é consistente. Em outras palavras, a ferramenta de coerência pode verificar se as validações, elementos e operações são corretamente relacionados e oferecem integridade. A ferramenta de coerência, em particular, é adaptada para ser executada usando a GUI. De maneira exemplificativa, um usuário pode definir e criar um ou mais conjuntos de código fonte usando a GUI, conforme mostrado na figura 25. Ao selecionar o botão 16, uma etapa de projeto ou associação para selecionar e associar as duas ou mais seções de código legível por máquina interrelacionado é realizada. De maneira vantajosa, um ou mais projetos/produtos podem ser escolhidos a partir da lista de produtos predefinidos disponíveis. De acordo com o exemplo mostrado na figura 25, o modelo "Core_Motor" é selecionado. O modelo "Core_Motor" pode representar uma definição de duas ou mais

seções selecionadas de código legível por máquina. Ao selecionar o botão 16, o código fonte do modelo "Core_Motor" é criado, em particular, as associações das duas ou mais seções de código legível por máquina são estabelecidas.

5 Como uma próxima etapa, que usa a dita GUI, que agora proporciona um botão 18, conforme mostrado na figura 26, a ferramenta de coerência é ativada, que agora verifica as duas ou mais seções de código legível por máquina selecionadas e associadas, isto é, a verificação implementada por computador das associações de dois ou mais códigos legíveis por máquina selecionados e associados é realizada. Em outras palavras, a coerência de duas ou mais seções selecionadas e associadas de código legível por máquina é verificada usando a ferramenta de coerência ao selecionar o botão 18. A ferramenta de coerência pode ser executada no fundo e proporcionar uma mensagem, por exemplo, quando a verificação de coerência for negativa, que particularmente proporciona detalhes para a natureza da coerência negativa. A ferramenta de coerência pode proporcionar também uma mensagem positiva, no caso de uma verificação positiva da coerência.

Também é possível que a ferramenta de coerência proporcione uma janela individual na GUI ou seja exibida como uma janela na GUI, conforme mostrado nas figuras 27a, 27b e 27c. Conforme mostrado nestas figuras, os parâmetros específicos podem ser selecionados, tais como, "coerência de Elementos", "coerência de Validações" e/ou "coerência de classificações". Também, a GUI/ferramenta de coerência pode ser adaptada para proporcionar detalhes da validação de coerência, conforme também mostrado nas figuras 27a, 27b e 27c.

5.2. Evitar problemas de formato

Dois exemplos são fornecidos a fim de descreverem esta vantagem:

Validações

30 é possível apresentar a seguinte validação definida:

'Se E1 > Op1, então'

onde:

E1= valor de elemento

Op1= resultado de um cálculo

Se o elemento E1 for definido com 'formato alfanumérico' e o resultado da Operação 1 for definido como 'formato numérico', o gerador não
 5 será capaz de comparar seus valores e, conseqüentemente, nenhuma mensagem de erro será criada pelo gerador que gera os arquivos executáveis interrelacionados. No tempo de execução, o processo irá falhar e uma descrição de erro será escrita no registro de problema.

De maneira vantajosa, a ferramenta de coerência pode verificar
 10 as inconsistências e produzir uma respectiva mensagem, mediante a qual a inconsistência pode ser corrigida. No exemplo, E1 será definido como 'Numérico'. Em outras palavras, mesmo se o gerador não proporcionar qualquer aviso e/ou mensagem de erro, a ferramenta de coerência é capaz de determine a inconsistência e produzir em texto claro um aviso e/ou mensagem de
 15 erro, mediante a qual a validação pode ser corrigida.

Em outras palavras, se a cláusula que compreende duas variáveis for uma seção de código. Ao verificar se as variáveis "E1" e "Op1" podem ser comparadas usando o operador matemático ">", a coerência da seção de código é verificada, em particular, se a dita seção de código é capaz
 20 de produzir uma saída consistente. Esta coerência pode ser particularmente verificada usando a ferramenta de coerência.

b) Operações

é possível ter a seguinte operação definida:

Operação 1 = Cálculo 1 + Cálculo 2

25 Cálculo 1 = Elemento 1 + Elemento 2

Cálculo 2 = Operação 2

onde:

a operação 1 é definida como: Dec (15, 3)

o cálculo 1 é definido como: Dec (15, 3)

30 o cálculo 2 é definido como: Dec (11, 6)

a operação 2 é definida como: Dec (15, 3)

Geralmente, estes resultados de operações que são importações

econômicas, são definidos como Dec (15, 3), enquanto as porcentagens são definidas como Dec (11, 6).

Neste exemplo, a Operação 2 é definida como Dec (15, 3), porém, o cálculo 2 é definido como Dec (11, 6). A operação 2 é definida com um formato numérico mais alto que o Cálculo 2. Neste caso, o gerador pode gerar um 'erro de Aviso'. A geração e a compilação irão terminar, porém, o usuário será avisado sobre este fato.

Durante o tempo de execução (também referido como tempo de operação), se todos os resultados da operação 2 se ajustarem ao formato definido para o Cálculo 2, nada irá acontecer. Porém, existe um risco, no caso de os resultados da operação 2 não se ajustarem ao formato de Cálculo 2 (ex: 123456789, 122). Neste caso, durante o tempo de execução, o sistema irá gerar um erro.

De maneira vantajosa, a fim de evitar um possível erro durante o tempo de execução, o formato do Cálculo 2 pode ser alterado para Dec (15, 3), em que o formato (15, 3) significa que o valor pode ter no máximo 3 decimais e 12 números inteiros; o formato (11, 6) significa que o valor pode ter no máximo 6 decimais e 5 números inteiros. Usando particularmente a ferramenta de coerência, o aviso criado pelo criador/compilador será traduzido e produzido em texto claro para o fomentador que não precisa saber a linguagem legível por máquina, na qual o gerador também produz o aviso. Consequentemente, o usuário também entende o aviso e pode adaptar o sistema ao corrigir a inconsistência. Em outras palavras, de acordo com o presente pedido, a coerência das associações das seções de código é verificada de maneira vantajosa, em que a "Operação 1" representa uma primeira seção de código e a "Operação 2" representa uma segunda seção de código e a associação entre as duas seções de código é realizada ao definir

Operação 1 = Cálculo 1 + Cálculo2,
onde "Cálculo 2 = Operação 2".

Também é possível que a "Operação 1" seja usada para gerar um arquivo executável. A "Operação 2" pode ser usada para gerar um arquivo executável adicional. Consequentemente, ambos os arquivos executáveis

são interrelacionados, uma vez que a saída de um arquivo executável (por exemplo, a saída da "Operação 2") é usada como a entrada do outro arquivo executável (por exemplo, da "Operação 1").

5.3. Evitar problemas de sistema

5 De maneira vantajosa, os parâmetros do compilador são adaptados e otimizados e o programa de classificação pode ser separado, de modo que o processador, mesmo quando tem a capacidade limitada (outros possíveis processos podem ser executados ao mesmo tempo), os arquivos executáveis interrelacionados podem ser gerados sem falha de geração. Em particular, o gerador pode ser separado em diferentes programas que atendem as funcionalidades (Operações, Validações, textos e documentação) proporcionando, deste modo, geração confiável dos arquivos executáveis interrelacionados.

15 Consequentemente, de acordo com o método acima, um conjunto de condições técnicas que será apresentado para as autoridades reguladoras de seguro e que não pode ser comercializado como tal, pode ser usado. As condições adicionais podem ser adicionadas ao dito conjunto de condições técnicas. Estas condições podem ser criadas para uma parte específica de uma organização, tal como, uma empresa, linha de negócio, área geográfica, canal de distribuição. Tudo isto pode ser aplicável, para ser um produto que pode ser definido e as ditas condições adicionais podem ser criadas ao seleccionar a partir do conjunto de condições técnicas, o que é e o que não é aplicável às condições adicionais.

25 As condições adicionais podem ser mapeadas em apenas um conjunto específico de condições técnicas e também podem restringir o conjunto de condições técnicas. As condições adicionais podem definir o produto, à medida que o mesmo é conhecido pelos clientes e vendido por intermediários. As condições adicionais podem ser específicas à quem as vende e/ou à quem as mesmas são vendidas:

- 30
- Descontos comerciais e/ou
 - Componentes de risco obrigatórios/opcionais e/ou
 - Limitações de elemento

A figura 28 mostra uma captura de tela exemplificativa de uma interface gráfica de usuário (GUI) 101. A GUI 101 compreende inúmeras entidades de entrada/saída, tais como, listas suspensas 110, botões 112, campos de entrada 114 e janela de entrada/saída 116. De maneira exemplificativa, mesmo se a GUI 101 for usada para proporcionar diferentes produtos, isto é, para criar apólices de seguro de carro e/ou apólices de seguro e/ou apólices de seguro de saúde, etc., as listas suspensas 110 e os botões 112 podem ser substancialmente mantidos. Também é possível que os campos de entrada 114 possam ser substancialmente mantidos. Como um exemplo, um campo de entrada 114a, o produto específico pode ser definido para o qual o texto legível por humano, por exemplo, uma apólice, deve ser criado. Na GUI exemplificativa 101, conforme mostrado na figura 28 como o produto específico, o seguro de construção é escolhido.

A janela de entrada/saída 116 pode ser particularmente usada para definir os parâmetros do produto, isto é, os parâmetros e/ou valores específicos da apólice de seguro de construção. De maneira exemplificativa, nove elementos de exibição 118a a 118i são mostrados. Os elementos de exibição 118a a 118i refletem os elementos repositórios, conforme definidos quando definem o produto e/ou quando definem a alteração do produto. Como um exemplo, no campo de entrada 114a, o termo "carro" pode ser introduzido, deste modo, como o seguro de carro de produto exemplificativo e como as apólices de seguro de carro de texto legível por humano exemplificativas.

Para produtos diferentes, isto é, diferentes tipos de apólices possíveis, os elementos da janela de entrada/saída 116 podem ser diferentes, o que significa que diferentes elementos de exibição podem ser mostrados em diversas janelas de entrada/saída 116. Deste modo, mais ou menos elementos de exibição podem ser exibidos e/ou pelo menos um dos elementos de exibição pode ser diferente. Como um exemplo, em vez do elemento de exibição 118c "tipo de construção de edifício", um elemento de exibição pode se referir ao "fabricante do carro".

Como pode ser visto a partir da figura 28, para cada elemento de

exibição 118a a 118i, proporciona-se um campo de entrada 120a a 120i. Os campos de entrada 120a a 120i podem ser usados a fim de escrever dados em uma tabela e/ou ler dados a partir de uma tabela. Como um exemplo, o campo de entrada 120a é usado para especificar o valor do elemento de exibição 118a "ano de construção de edifício" a "inferior ou igual a 1940". Em outras palavras, é possível que quando definir o produto, isto é, quando os arquivos executáveis são criados, um elemento repositório que pode ser introduzido é o "ano de construção de edifício". Em uma tabela relacionada, um elemento de tabela que pode ser definido mantém o valor do dito elemento repositório. Neste caso exemplificativo, o valor é "inferior ou igual a 1940"

Na GUI exemplificativa 101 conforme mostrado na figura 28, os campos de entrada 120a a 120e podem ser preenchidos apenas usando valores predefinidos. Entretanto, também é possível que os campos de entrada não tenham quaisquer valores predefinidos, mas, de preferência, possam ser preenchidos de maneira individual, conforme mostrado nos campos de entrada 120f a 120i. Em outras palavras, os campos de entrada 120a a 120i ou os valores dos campos de entrada 120a a 120i, que podem ser introduzidos manualmente por um usuário ou ser recuperados semiautomaticamente, por exemplo, por meio de uma lista suspensa ou automaticamente, podem ser referidos como conteúdo dos respectivos elementos de exibição 118a a 118i.

Além disso, a GUI 101 compreende uma barra de rolagem 122. Portanto, a janela de entrada/saída 116 pode compreender inúmeros elementos de exibição adicionais 118 e, conseqüentemente, campos de entrada adicionais 120, que não são mostrados na figura 28.

A GUI 101 refere-se ao proporcionamento de uma nova cotação, conforme pode ser visto a partir do cabeçalho 124. Portanto, o texto legível por humano que é criado de acordo com o método descrito acima é uma cotação de uma taxa de uma apólice de seguro definida. Conforme mostrado na figura 28, enquanto a GUI 101 refere-se ao estabelecimento de uma nova cotação, por exemplo, para um seguro específico, uma GUI similar pode ser usada para outros propósitos, em que o esboço visual da GUI se mantém

substancialmente. Isto é mostrado de maneira exemplificativa na GUI 101 na figura 29.

Conforme pode ser visto na figura 29, a lista suspensa 110 e os botões 112 são idênticos à GUI 101, conforme mostrado na figura 28. A GUI 101 compreende um cabeçalho 126 que indica que a GUI 101 refere-se à criação de uma nova apólice. Consequentemente, a janela de entrada/saída 116 compreende elementos de exibição 118a' e 118b'. Consequentemente, para cada elemento de exibição 118a' a 118b', existe um campo de entrada correspondente 120a', 120b'. Conforme mostrado na figura 29, ambos os campos de entrada 120a', 120b' ainda se encontram vazios, porém, podem ser preenchidos, por exemplo, por um usuário da GUI e/ou automaticamente.

Consequentemente, proporcionou-se uma interface muito eficiente e facilmente entendível para proporcionar um grande numero de diferentes textos legíveis por humano. Tal texto legível por humano pode ser uma cotação de apólice de seguro e/ou uma própria apólice de seguro e/ou partes de uma apólice de seguro. Entretanto, a dita interface, isto é, a GUI 101, é tão flexível, que a mesma permite a adaptação do ambiente de computação subjacente, por exemplo, a fim de introduzir produtos adicionais. No entanto, o esboço visual se mantém substancialmente inalterado, de modo que o usuário da GUI sem ter substancialmente que requerer qualquer conhecimento adicional de uso e/ou manutenção, possa acessar e usar uma GUI mesmo quando o ambiente de computação tiver sido adaptado.

A figura 30 mostra um fluxograma adicional que proporciona uma visão geral de etapas de processo que podem ser realizadas, em particular, quando se adaptam ao ambiente de computação.

Na etapa S1', as definições de requerimentos que podem ser proporcionadas atendem as necessidades do usuário do ambiente de computação. Em particular, as ditas definições de requerimentos podem compreender, como texto legível por humano e/ou instruções, as alterações que foram feitas no ambiente de computação. Tais definições de requerimentos podem ser tão simples quanto a definição de que o ambiente de computação possa ser capaz de gerar um novo produto. As definições de requerimentos

que podem ser adicionalmente detalhadas compreendem, em particular, todos os elementos que o novo produto precisa cobrir. Como um exemplo, a definição do requerimento pode compreender que o novo produto refere-se ao seguro de saúde e a definição de requerimentos também compreende

5 que os parâmetros/variáveis específicos precisam ser ajustados pelo usuário da GUI. Tais parâmetros podem ser de gênero, idade, pessoa, etc.

Na etapa S2', a definição de produto é feita. A definição de produto compreende o conjunto completo de parâmetros/variáveis que são necessários, a fim de cumprir os requerimentos, conforme definido na etapa

10 S1'.

Na etapa S3', o produto é configurado. A configuração do produto pode compreender particularmente a definição do produto no texto legível por máquina. Em outras palavras, conforme descrito acima, o código do produto pode ser configurado, em que tal configuração pode ser substancial-

15 mente realizada em texto legível por humano, conforme definido acima.

Na etapa S4' os arquivos executáveis são gerados. Conforme descrito acima, tal geração é particularmente realizada ao compilar o código legível por máquina no lado de servidor.

Na etapa S5', o software é instalado no lado de servidor. Isto pode

20 de compreender que o software é instalado em um motor servidor ou em um ou mais computador de clientes, conectados ao motor servidor.

Na etapa S6', um teste do produto é realizado pelo usuário. Este teste pode ser particularmente realizado pelo usuário que também fez a definição de requerimentos na etapa S1'.

Na etapa S7', verifica-se se o usuário aceita o produto. No caso afirmativo, o método é finalizado na etapa S10'. De outro modo, na etapa S8', se decide se a configuração de produto precisa ser alterada. Isto pode ser particularmente o caso, se os parâmetros/variáveis precisam ser alterados e/ou introduzidos e/ou deletados. A razão para isto pode ser que as regras de negócio adicionais podem ser cobertas e/ou que as regras de negócio

25 30 alteraram. Neste caso, o método é repetido a partir da etapa S3'.

De outro modo, isto é, se nenhuma alteração da configuração de

produto for necessária, a partir da etapa S8' quaisquer novos requerimentos podem ser implementados na etapa S9'. Particularmente, tal implementação não compreende nenhuma alteração dos arquivos executáveis. Como um exemplo, tal implementação pode compreender, por exemplo, corrigir a GUI
 5 ao corrigir partes da aparência visual da GUI. Especificamente, o termo nome de empresa no campo 118a' na figura 29 pode ser substituído pelo termo "parte de negócio".

Após a implementação de todos os requerimentos na etapa S9', o método é finalizado na etapa S10'.

10 Como um resumo, as etapas S1', S4', S5', S6', S7', S8' e S9' podem ser realizadas no lado de servidor ou em qualquer computador adicional que é conectado no lado de servidor. As etapas S2' e S3' são realizadas em um lado do cliente.

15 Fornecida a descrição acima, o presente pedido pode permitir a realização do seguinte procedimento em partes ou no total. Deste modo, também é possível que partes do procedimento sejam realizadas de maneira repetida e/ou que partes do procedimento ou todo o procedimento seja realizado em ordem diferente.

20 O método, conforme descrito neste pedido, proporciona de maneira vantajosa uma solução unificada para muitas, particularmente, todas as transações de apólice (novos negócios, endossos) e pra muitos, particularmente, todos os tipos de produtos, tal como, Seguro Geral e/ou Automóvel e Propriedade & Acidente e/ou Vida & Pensões, etc.

25 Este método exemplificativo pode compreender funções comuns para muitos ou todos os produtos:

Cotações

Novos Negócios e modificações de apólice (endossos, de volta à data & endossos temporários,...)

Cancelamentos de Apólice

30 Restabelecimentos de Apólice

Triagem de apólice

Renovação de Portfólio

Adicionalmente, o método também pode compreender as funções específicas para alguns tipos de produtos, tal como, Vida & Pensões, Automóvel, Propriedade & Acidente.

De maneira vantajosa, o método descrito permite a definição de produto através dos usuários especialistas com a mínima intervenção de departamento IT.

Ademais, de maneira vantajosa, o projeto e a funcionalidade de sistemas podem permitir a criação e a modificação de um produto de uma maneira rápida e pouco dispendiosa, especificamente, as telas de Administração de Apólice podem ser automaticamente geradas a partir da definição de produto na Fábrica de Produto.

De maneira vantajosa, os processos consistentes para apólices individuais e/ou de múltiplos objetos e/ou grupo podem ser providos.

Além disso, de maneira vantajosa, uma metodologia comum é definida para criar produtos em uma empresa de seguro que permite, em particular, a determinação dos limites de risco e condições financeiras que uma empresa deseja encarar/oferecer em sua atividade de seguro.

A seguir, descreve-se um exemplo de ciclo de vida de uma apólice, em que as sucessivas etapas são descritas e o ciclo de vida pode começar em qualquer uma das ditas etapas:

Etapas: Solicitação/Cotação

O cliente entra em contato com a empresa e solicita a informação de cobertura de seguro. Uma cotação é fornecida sem cobertura de risco.

Etapas: Novo Negócio

Registro de um novo negócio na base de um produto, uma apólice, uma cotação ou um modelo é realizado.

Etapas: Cancelamento

Se a apólice estiver ativa, o cancelamento da apólice pode ocorrer em qualquer ponto na vida da apólice.

Etapas: Modificações/Endossos

Esta etapa pode se referir aos endossos econômicos e/ou en-

dossos não econômicos.

Etapa: Reintegração

A reativação de uma apólice inativa é realizada.

Etapa: Portfólio

5 A renovação de apólices atuais é realizada.

A ordem acima das etapas de método pode continuar com a etapa de Solicitação/Cotação.

A seguir, detalhes adicionais das etapas acima são proporcionados.

10 Etapa: Solicitação/Cotação

A nova função de quota que pode ser incorporada na função de solicitação/cotação permite que o usuário lance os dados de data, deste modo

15 o sistema oferece diversas opções de classificação e/ou cotações são automaticamente armazenadas e numeradas quando todos os dados necessários são lançados (identificação cliente, objeto, etc.) e/ou

as cotações podem ser feitas a partir de cotações e apólices existentes, ou ser baseadas em um produto ou um modelo e/ou

20 os dados requeridos para uma nova cotação são inferior aos dados requeridos para um novo negócio e/ou

a cotação de diversas opções de seguro e pagamento no mesmo produto é permitida.

25 A função de cotação a apólice, que pode ser incorporada na etapa solicitação/cotação permite selecionar uma cotação existente e solicitar um novo negócio. O sistema irá abrir o novo negócio ao transferir todos os dados de cotação para a forma de aplicação. Se uma cotação tiver sido feita para diversas opções, o sistema irá solicitar uma única escolha para o novo negócio.

30 O termo "modelo" refere-se a um esboço de apólice a partir do qual as apólices são criadas. O mesmo inclui detalhes do objeto segurado e da classe prêmio retirado. Seu uso principal serve para processar novos ne-

gócios de maneira rápida sem ter que lançar detalhes da escolha de classe prêmio ou do objeto segurado.

Etapa: Novo negócio

De maneira vantajosa, no método descrito, o mesmo fluxo de janelas pode ser usado para todas as configurações de novo negócio qualquer que seja o produto. Portanto, método é facilmente utilizável pelo usuário.

Registros de novo negócio podem ser feitos na base de Produto, Cotação, Modelo, Apólice. De maneira vantajosa, o registro de novo negócio pode ser possível na base apólice, cotação ou modelo. Portanto, pó processo de emissão de apólice é mais fácil porque os dados pré-armazenados podem ser usados. As condições de produto especiais podem ser definidas para alguns clientes, negócios ou intermediários.

De maneira vantajosa, o método acima proporciona um fluxo de navegação exclusivo e as mesmas telas podem ser usadas no processo de registro de um novo negócio.

As seguintes informações/dados podem ser usados para for definir o novo negócio:

A duração de apólice pode ser cancelável/temporária/decenal e/ou

Datas de início e término e/ou
nível de emissão de recibo e/ou
forma de pagamento e/ou
tipo de cosseguro ou resseguro e/ou
tipo de regularização e/ou
detentor de apólice e coletor de dados e/ou
dados de intermediários e comissões e/ou

Cosseguro e resseguro no novo negócio

Se uma Police contém dados de cosseguro ou resseguro, estes dados podem ser lançados junto com os detalhes de apólice gerais. Quando os detalhes de cosseguro e resseguro são lançados, os mesmos podem ser especificados se forem cosseguros aceitos ou transferidos, e/ou resseguros

compulsórios e/ou opcionais.

No caso de cessões de cosseguro, as empresas às quais parte do risco é cedido podem lançar as seguintes informações:

- 5 % de participação da empresa de seguro e/ou
nome/nome registrado da empresa/empresas que aceitam o risco e/ou

- % de participação de empresa de aceitação e/ou
porcentagem do consórcio que corresponde à empresa de aceitação, que é automaticamente lançada em 4% e/ou
10 porcentagem de despesas administrativas da apólice estabelecida pela empresa que faz a cessão. Isto é automaticamente lançado em 1%.

Um único recibo ou um recibo para cada empresa pode ser emitido quando a apólice é uma cessão de cosseguro.

- 15 No caso de cosseguro aceito, os seguintes detalhes sobre a empresa no comando podem ser lançados:

- Participação da empresa de seguro e/ou
 % de consórcio a ser paga pela empresa de seguro e/ou
 % de despesas administrativas no gerenciamento de apólice estabelecido pela empresa de transferência e/ou
20 % de comissão paga pela empresa de seguro ao intermediário e/ou

 nome/nome registrado da empresa responsável pelo risco e/ou
 o numero de apólice, conforme designado pela empresa no comando.

- 25 Controle de apólice (como parte da etapa: Novo negócio)

A tela de controle de apólice é estrutura de maneira exemplificativa da seguinte forma:

- Cabeçalho que identifica a apólice e/ou
 árvore de apólice e/ou
30 botões relacionados à árvore de apólice: Novo objeto, modificação de objeto, seleção de classe prêmio, modificação de classe prêmio, etc e/ou

outras funções, tais como, configuração de beneficiários de apólice, cláusula,...

A função de novo objeto segurado pode ser acessada a partir da tela mencionada acima. O número de objetos segurados determina o tipo de apólice. Em Solução de Seguro esta poderia ser:

Individual/Coletivo, por exemplo, como uma função do número de objetos segurados e/ou

Múltiplos objetos, por exemplo, uma apólice pode ter mais do que apenas um objeto segurado. Por exemplo: um carro e uma casa e/ou

Múltiplas situações, por exemplo, uma apólice pode ter mais de um caso de um objeto com diferentes riscos, como um exemplo, 2 carros podem ser compreendidos na mesma apólice.

Objeto segurado (como parte da etapa: Novo negócio)

Uma tela de novo objeto segurado geral, similar à captura de tela da GUI 101, conforme mostrado nas figuras 28 e 29, pode ser exibida se nenhuma tela de objeto específico já existir. Uma tela de novo objeto segurado específico relacionada ao tipo de objeto que é segurado pode ser mostrada se esta tela específica já existir. As telas específicas podem ser desenvolvidas para aqueles produtos com objetos que requerem desenvolvimento específico, a fim de registrar a informação de objeto. Por exemplo, se o objeto que é segurado for um carro, a informação de proprietário/motorista será requerida como elementos de exibição exemplificativos. Entretanto, se o objeto que é segurado for uma casa, a informação de risco de elementos de exibição exemplificativos, tipo de casa, etc. será requerida. E para uma informação de produto Unit Link sobre a pessoa que é segurada será requerida como elementos de exibição exemplificativos. A entrada pode ser o elemento de tabela específico que pode ser consequentemente preenchida.

Classe prêmio (como parte da etapa: Novo negócio)

A seleção de classe prêmio pode ser feita usando uma GUI específica, em que a classe prêmio para o produto comercial selecionado pode ser mostrada, tal como a GUI. É possível que uma classe prêmio ou um pacote de classe prêmio possa ser selecionado. Uma vez selecionadas, as

classes prêmio podem ter valores designados às mesmas.

Valores (como parte da etapa: Novo negócio)

Os tipos de classe prêmio dependem de qual tela de produto é mostrada. A tela específica (também referida como GUI) pode mostrar a entrada do valor da classe prêmio relacionada a este produto.

De maneira vantajosa, as telas específicas foram desenvolvidas para certos produtos. Entretanto, uma tela padrão encontra-se disponível para diversos produtos que têm valores diferentes dependendo do produto. A tela pode ter um ou mais dos seguintes campos:

Classe prêmio, por exemplo, que mostra a classe prêmio selecionada que foi combinada no contrato.

Elementos, por exemplo, que contêm uma descrição dos elementos que descrevem cada classe prêmio.

Valores, por exemplo, que contêm os campos para cada elemento da classe prêmio.

Similares às capturas de tela, conforme mostrado nas figuras 28 e 29, os elementos de exibição e/ou os campos de entrada podem ser definidos ou compreender um ou mais dos campos mencionados acima.

Árvore de Apólice (como parte da etapa: Novo negócio)

A árvore de apólice mostra a relação entre os elementos existentes em uma apólice, em particular, exibindo um ou mais:

Apólice que é um novo negócio.

Tipos de objetos que podem ser segurados pelo produto.

Os objetos segurados na apólice agrupados na árvore pelos tipos de objeto.

Classe prêmio através da qual cada objeto é segurado. Na árvore de apólice, a classe prêmio coberta na apólice é mostrada para cada objeto segurado.

Estes elementos podem ser consultados, modificados e, no caso do objeto segurado, os elementos podem ser duplicados e cancelados.

Resultados de Classificação (como parte da etapa: Novo negócio)

Uma tela de resultado de classificação mostra o prêmio que o

cliente irá pagar. Ademais, o período de prestação é um programa que determina como dividir o prêmio em inúmeros recibos, tanto de acordo com a forma de pagamento selecionada no prêmio de movimento anual, como na informação proporcionada pela Fábrica de Produto. Os detalhes dos prêmios resultantes podem ser acessados.

Período de Prestação (como parte da etapa: Novo negócio)

As seguintes validações podem ser realizadas pelo período de prestação:

Itens Fracionais: Isto determina quais itens devem ser cobrados no primeiro recibo e quais itens devem ser divididos entre os recibos subsequentes (uma taxa sendo um exemplo de um item que não pode ser dividido e deve ser cobrado no primeiro recibo).

Itens reembolsáveis: No caso em que uma classe prêmio é subscrita, ou uma apólice cancelada, esta função determina se um item pode ser devolvido para o cliente ou não (o consórcio sendo um exemplo de um item não reembolsável em produtos de vida).

Itens não Pró-tributáveis: Isto determina se a anuidade total deve ser coletada mesma quando o prêmio não é retirado em uma base anual (um exemplo é a cobertura de viagem de automóvel).

Beneficiários (como parte da etapa: Novo negócio)

A solução de seguro permite que diferentes beneficiários sejam designados a diferentes níveis da apólice, ou seja, apólice e/ou objeto e/ou classe prêmio.

Para designar beneficiários, um elemento da árvore de apólice pode ser selecionado na tela de Controle de Apólice e, então, o botão 'Beneficiários' é selecionado. Após isto a tela de Manutenção de Beneficiários é acessada.

Comissões (como parte da etapa: Novo negócio)

Se a comissão selecionada a partir da folha de intermediários na tela de dados de apólice individual for 'Referida' ou 'Distribuída', então, o sistema não permite a emissão da apólice até as comissões serem configuradas na tela 'Comissões Manuais'. Esta tela pode ser obtida usando-se o bo-

tão 'Comissões (a partir da tela de Controle de Apólice)

Documentação (como parte da etapa: Novo negócio)

As condições particulares associadas a um produto que é subscrito e os parágrafos que formam o documento podem aparecer como texto na janela de composição do documento. Todo o texto pode compreender parágrafos. Os parágrafos pode ser

Compulsórios, por exemplo, o(s) parágrafo(s) são necessários para o texto ou

Não compulsórios, por exemplo, o(s) parágrafo(s) não são necessários para o texto.

Além disso, os seguintes parágrafos podem ser distinguidos:

Parágrafos fixos: O usuário não pode modificar, adicionar ou eliminar qualquer texto no parágrafo. Alguns parágrafos fixos têm "variáveis" que o usuário deve preencher.

Parágrafos livres: Alguns textos oferecem este tipo de parágrafo, que permite que o usuário inclua parágrafos com conteúdo que é totalmente lançado pelo usuário.

Quando todos os parágrafos do texto são concluídos, é possível exibir a primeira retirada do documento antes de salvar os dados. Para isto, o botão 'Pré-visualização' pode ser ativado. Após verificar que todas as condições exibidas estão corretas, o dado é salvo através da ativação do botão de janela global 'Aceitar'.

Além disso, a impressão da documentação fixada na apólice ou endosso pode ser permitida de maneira centralizada e/ou distribuída. De maneira alternativa ou adicional, a documentação pode ser enviada por fax e/ou e-mail, sem sair do pedido.

Administração de Competência (como parte da etapa: Novo negócio)

A administração de competência é uma função que estabelece limites aos usuários quando contratam uma apólice, que cria uma cotação ou marcação de um endosso. O sistema pode gerar incidências de administração de competência quando, durante o processo de criação de uma apólice, cotação ou endosso, o usuário realiza uma ação que excede as competên-

cias as quais ele foi designado.

Como um exemplo, quando o usuário está desenvolvendo uma operação de subscrição (apólice, endosso, etc.) e uma incidência relacionada a uma competência é gerada, a operação não pode ser emitida até que as causas da incidência sejam corrigidas ou autorizadas. O processo pode ser concluído, uma vez que todos os erros foram autorizados. Uma modificação que pode ser requerida refere-se ao motivo do erro (por exemplo, capital em excesso). A emissão não será concluída até quaisquer erros não autorizados sejam modificados.

10 A administração de competência pode compreender as seguintes etapas:

I. Um usuário emite uma apólice com um capital maior do que ele é autorizado.

15 II. Uma incidência de administração de competência que é gerada evita a emissão ser continuada.

III. Usando notas, a autorização é solicitada para o usuário responsável ou para o usuário com nível de autorização adequado.

IV. O usuário pode ou não autorizar a emissão da apólice ou endosso através da operação de autorização de Apólice.

20 V. Usando notas, o usuário que estava desenvolvendo a operação é informado se ele pode ou não continuar.

Seleção de Saída (como parte da etapa: Novo negócio)

O método descrito pode permitir que a documentação ligada à apólice ou endosso seja impressão de uma maneira centralizada ou distribuída. Isto também permite que a documentação seja enviada por fax ou correio eletrônico sem sair do pedido. Uma vez que o tipo de saída para uma documentação específica foi selecionada, o processo de emissão irá terminar.

Etapa: Emissão

30 Um processo de emissão pode consistir em uma série de subprocessos comuns e outros subprocessos que serão executados de acordo com os parâmetros fornecidos. Os processos exemplificativos que são co-

muns a todos os produtos e as etapas de emissão (novo negócio, endossos,...) podem ser:

Emissão de recibos e/ou

Consolidação de tabelas de novo negócio e/ou

5 Documentação

Os subprocessos exemplificativos que podem ser executados de acordo com os parâmetros fornecidos podem ser:

De acordo com o produto e SBU

Acidente: Lista de segurados e/ou

10 Empresas e SMEs: Folha PML /ou
diversos: FIC.

De acordo com o movimento de cancelamento de etapa: FICO.

Etapa: Endossos

15 A criação de endossos pode permitir que modificações sejam incluídas na apólice ou em qualquer um de seus elementos ou entidades. Os endossos podem ser

Endossos não econômicos: o prêmio não é modificado e/ou

endossos econômicos: envolvem uma modificação do prêmio. O sistema compara o prêmio a partir do movimento de base com o novo endosso. Se existem diferenças no prêmio, o endosso é um endosso econômico.

20 De preferência, o fluxo de um endosso é o mesmo que aquele de um novo negócio. Tipicamente, pode ser necessário identificar a razão por trás de uma modificação em um endosso. Uma tabela de corporação pode
25 listar as razões e pode designar os valores de acordo com o produto de apólice. Dependendo da razão, certos campos de tela podem ser ativados ou desativados. Cada modificação na apólice cria uma nova versão, porém, nem todas as modificações envolvem um novo endosso. O tipo e a razão por trás da modificação irá determinar se a modificação gera um endosso ou
30 não.

Um novo endosso será gerado se a modificação requer que o detentor da apólice receba a nova documentação. O sistema irá comparar os

prêmios de movimento de base com aqueles do novo endosso. Qualquer prêmio diferencial irá resultar em um endosso econômico (oposto aos endossos não econômicos). Um endosso deve especificar uma razão identificada para a modificação. Uma tabela de corporação lista as razões e designa os valores de acordo com o produto de apólice. As razões podem ativar ou desativar certos campos de tela.

Data de efeito, o vencimento (em efeito, após a renovação de portfólio) ou data de endossos de cobertura de apólice endossos é permitida. Se os endossos forem desejados para uma data antes daquela de outro endosso, eles devem ser fragmentados em tantas partes quanto existem versões de apólice. Os endossos de extensão de apólice são permitidos para apólice temporária. As seguintes etapas podem ser realizadas:

I. A emissão de uma nova apólice anual, por exemplo, Apólice 1234567890 Versão 0 Endosso 0.

II. Se o detentor de apólice deseja incluir uma nova classe prêmio, em efeito 1/06/2002, um endosso é operado em efeito 01/06/2002, por exemplo, Apólice 1234567890 Versão 1 Endosso 1.

III. Se um erro na comissão intermediária foi descoberto, 2 endossos são operados. O primeiro irá corresponder à versão 2 Endosso 1 e o segundo à Versão 3 Endosso 1.

IV. Se cliente deseja pagar por prestações em efeito a partir do próximo ano, um endosso será operado no vencimento, que ocorre mediante uma renovação da apólice, por exemplo, Apólice 1234567890 Versão 4 Endosso 2.

Um endosso é operado na última versão da apólice ou em qualquer versão anterior. Quando operado na última versão, o movimento atualizado será aquele que existe atualmente no portfólio de apólice.

A classificação também é fixa na Fábrica de Produto. Os recibos e rendimentos são emitidos com base nos resultados gerados pela fábrica. O prêmio anual positivo é deduzido do prêmio pro rata negativo do movimento anterior.

As modificações podem ser registradas em todos os casos, en-

tretanto, aqueles além de seu nível de autorização não podem ser emitidos e ficarão pendentes até a autorização ser fornecida a partir do nível adequado. A funcionalidade do sistema de autorizações é idêntica àquela que opera para o novo negócio.

- 5 Ao chamar a Fábrica de Produto, quando solicitar emissões de endosso, todas as validações de produto, competência e subscrição serão realizadas. A Fábrica de Produto irá proporcionar os erros mediante os quais a emissão pode ou não ser possível. Quando um endosso é emitido, os ajustes nas cessões de resseguro e cosseguro serão feitos se os resultados de
- 10 pedido de endosso em uma variação nas cessões. Um endosso não pode ser emitido se as cessões permanecerem pendentes (resseguro facultativo não definido,)

Etapa: Manutenção de apólices pendentes

- 15 O estado de uma nova apólice ou de um endosso pode ser concluído ou pendente. Um movimento que ainda não foi emitido é um movimento pendente. A administração de apólice exemplificativa tem 2 bancos de dados: uma para as apólices completas e um para as apólices incompletas. A emissão de apólice pode registrada em um banco de dados de apólices incompletas até a emissão ser concluída. Os dados consolidados, então,
- 20 podem ser armazenados no banco de dados de apólices completos.

- Quando um movimento em uma apólice existente é feito (por exemplo, um endosso) este pode ser armazenado no banco de dados de apólices incompletas, enquanto o número de apólice é mantido. O usuário que deseja recuperar uma apólice por meio desta função deve ter atingido a
- 25 etapa de controle de apólice. Se os usuários desistirem antes, a apólice não pode ser recuperada. A apólice será bloqueada sempre que a mesma tiver um movimento pendente (por exemplo, um endosso não emitido) e irá permanecer bloqueada até a emissão do movimento pendente ser concluída. As apólices irão permanecer pendentes quando mostrarem erros de gerenciamento de competência e até o erro ser autorizado. A apólice, então, será
- 30 recuperável a partir de manutenção de apólices pendentes.

Etapa: Cancelamento

Esta função/etapa permite o cancelamento/anulação de uma apólice válida. As razões para a anulação são definidas para cada produto/objeto. Algumas razões são comuns a todos os produtos e outras são específicas. À medida que os motivos para o cancelamento podem variar.

- 5 Os cancelamentos de apólice são permitidos em qualquer data dentro da validade da apólice:

No caso de recibos pendentes, é possível cancelar as apólices com ou sem retorno. No caso de cancelamento com retorno, a porção não usada da apólice prêmio será reembolsada.

- 10 As apólices podem ser canceladas na data de vencimento. Neste caso, nenhum prêmio é reembolsado e a apólice retém sua cobertura – para propósitos de reivindicações – ao longo de sua validade e sem transferência para o portfólio.

- 15 Uma vez que todos os dados da apólice cancelável identificada foram lançados, a mesma pode ser diretamente cancelada ou os dados de classificação de movimento podem ser acessados a fim de se obter qualquer quantia de retorno existente. Finalmente, todos os documentos de cancelamento são gerados.

- 20 De maneira vantajosa, o método acima proporciona capacidades de reescrita "fora da caixa". Esta funcionalidade pode incluir a capacidade de ter uma nova apólice (substituição) subscrita no lugar de uma apólice antiga (aquela que é substituída). O processo pode envolver anulações através dos usuários (cancelamento da apólice que é substituída) e/ou novo negócio (subscrição de uma apólice de substituição).

- 25 Etapa: Reintegração

- 30 As reintegrações permitem que as apólices previamente canceladas tenham suas validades restabelecidas. As apólices podem ser concluídas, validadas e registradas a fim de serem restabelecidas. A data de reintegração geralmente é a mesma que a do cancelamento efetivo da apólice, porém a mesma pode ser em um período posterior.

A reintegração pode envolver o pagamento de lucros, a restauração de resgates ou o pagamento de interesse através do retentor de apóli-

ce, a fim de restabelecer a apólice na mesma condição em que a mesma foi cancelada. Em todos estes casos as validações necessárias podem ser realizadas, assim como as classificações (Fábrica de Produto) para a avaliação do novo prêmio.

- 5 O processo de restabelecimento de apólice também pode ser acompanhado por modificações de apólice. Estas modificações podem ser feitas na mesma operação de restabelecimento sem quaisquer endossos adicionais.

10 Finalmente, todos os documentos de restabelecimento (notas de restabelecimento, recibos, etc.) são emitidos e enviados.

Etapa: Triagem de Apólice

- 15 A triagem de Apólice permite que o portfólio de apólice seja revisado com base na frequência de acidentes e critérios de rentabilidade negativa. Uma análise de processo em lote das apólices de portfólio pode ser realizada com base nas tabelas/critérios predefinidos, que levam a detecção de necessidades de reorganização. O sistema, então, gera um (suporte) banco de dados de apólices que precisa de retificação com seus critérios de retificação correspondentes, e designa uma solução proposta genérica. Os usuários analisam a partir do banco de dados cada apólice individual e propõe
- 20 medidas de retificação específicas para cada uma delas.

Uma vez que as apólices com necessidade de retificação são detectadas, os usuários podem informar o cliente/intermediário da decisão de saneamento definitiva a ser executada:

- 25 **Manutenção:** A apólice mantém seu estado atual por diversas razões, por exemplo: bom desempenho global de cliente, reivindicações, etc.

Correção: Um endosso na apólice real é criado modificando-se alguns de seus detalhes, por exemplo: aumento de prêmios, alteração de condições de cobertura, etc.

- 30 **Cancelamento:** O sistema irá cancelar a apólice automaticamente, uma vez que a triagem de apólice é concluída e a decisão de cancelamento é tomada. Todo mês um processo em lote cancela todas as apólices que os usuários decidiram cancelar.

Etapa: Portfólio

Renovação de Portfólio Manual (como parte da etapa: Portfólio)

Esta função permite a renovação de apólice manual, sem ser incluída no processo de portfólio automático. Os usuários podem selecionar aquelas apólices que eles desejam renovar em uma base individual. Os usuários podem selecionar e/ou ajustar os prêmios a serem renovados. Todos os tipos de modificações são permitidos: alteração de taxa, cancelamento de classe prêmio, etc. Além do recibo, o sistema irá emitir o endosso correspondente através da ligação com o processo de endossos. Se a modificação foi registrada antes do processo de portfólio, ou seja, porque existe o endosso futuro (efetivo na data de vencimento de apólice), o usuário precisa apenas decidir sobre a renovação de condições registradas.

O portfólio manual não realiza a atualização como no processo de renovação automática, tais como, bônus/alteração de penalidade de nível, ações de renovação de portfólio, avaliação de somas ou seguradas ou prêmios, etc.

Renovação de Portfólio Automática (como parte da etapa: Portfólio)

O processo de portfólio é um processo de renovação mensal para todas as apólices que vencem no mês específico (parâmetros estabelecidos pelo produto), geralmente no mês seguinte do processo.

O processo envolve

- a atualização de intermediários, penalidade bônus, avaliação e renovação de portfólio e/ou
- a classificação dos prêmios da próxima anuidade e/ou
- a emissão da apólice com novas datas de renovação e/ou
- a impressão das notas de vencimento.

Seleção de Apólice (como parte da etapa: Portfólio/Renovação de Portfólio)

A validação de Fábrica de Produto para produtos com apólices de renovação automática. As apólices selecionadas podem ser

- em vigor e/ou
- com a data de expiração conforme mostrada no parâmetro e/ou
- com renovação automática.

A apólice é marcada com um aviso para os usuários de que a renovação se encontra em andamento. A apólice é bloqueada e nenhum movimento pode ser feito.

Atualização de valor (como parte da etapa: Portfólio/Renovação de Portfólio)

5 Esta etapa pode compreender uma ou mais:

Atualizações Intermediárias e/ou

Penalidade Bônus e/ou

Ações de renovação de portfólio e/ou

Avaliação e/ou

10 Renovação de datas.

Classificação (como parte da etapa: Portfólio/Renovação de Portfólio)

As apólices prêmio são obtidas na próxima anuidade, após os valores serem estimados pela Fábrica de Produto executável. Alguns produtos têm um prêmio mínimo definido na base das seguintes opções:

15 Prêmio mínimo por apólice e/ou

Prêmio mínimo por objeto e/ou

Prêmio mínimo por classe prêmio e/ou

Prêmio mínimo combinado (por objeto e classe prêmio).

20 Para as apólices que não excedem o prêmio mínimo, a diferença entre todas as apólices prêmio serão divididas (proporcionais aos seus prêmios), ou sua totalidade será aplicada à primeira classe prêmio, as conforme indicado pelo Fábrica de Produto.

Emissão de Apólice/Apólices (como parte da etapa: Portfólio/Renovação de Portfólio)

25 Este é um processo de validação e consolidação (tanto para apólices completas como para apólices incompletas) da atualização de dados da renovação de apólice. Uma série de rotinas em uma estrutura modular é requerida para o processo de emissão. Cada rotina fica encarregada da validação ou consolidação.

30 A mesma também inclui dois módulos adicionais que

Estimam as comissões intermediárias e/ou

Lançam dados de apólice na interface de recibos.

Emissão de Apólice/Green Cards (como parte da etapa: Portfólio/Renovação de Portfólio)

O coletor intermediário do primeiro recibo define a inclusão do green card na renovação de portfólio.

5 Emissão/Recibos de Apólice (como parte da etapa: Portfólio/Renovação de Portfólio)

Um modelo de dados de recibos é concluído com os dados fornecidos pela interface. Os pagamentos são fracionados em trimestres, semestres ou pagamentos anuais únicos, conforme indicados na apólice.

10 Os seguintes recibos são emitidos:

Pagamentos anuais e/ou

Primeiro pagamento trimestral ou semestral e/ou

Pagamentos sucessivos de apólices previamente expiradas que vencem no mês atual.

15 O restante dos recibos é mantido como 'Emissão Pendente'. Um arquivo Postscript (*.ps) é criado para armazenar todas as informações em recibos emitidos e serem redirecionadas para a impressora central da empresa. Para a facilidade de distribuição, a informação é classificada por escritório e intermediário.

20 Notas de Vencimento (como parte da etapa: Portfólio/Renovação de Portfólio)

O processo pelo qual toda a informação é coletada para completar as notas de vencimento. A nota informa o cliente dos prêmios aplicados na renovação de apólice.

25 Reconciliação (como parte da etapa: Portfólio/Renovação de Portfólio)

O processo extrai todos os dados atualizados no processo de renovação de apólice automática e transfere os mesmos para um arquivo Excel. O arquivo mostra uma falha através de SBU da seguinte informação:

Apólices renovadas totais e/ou

30 Total faturado e/ou

Total faturado para pagamentos de emissão pendentes devidos no mês atual e/ou

Porcentagem de comissões de Intermediários sobre o faturamento total e/ou

Total de prêmios para cosseguro.

5 Uma lista de apólices não renovadas acompanhadas por uma razão para tal:

Canceladas e/ou

Não renováveis e/ou

Bloqueadas no início do processo devido a um movimento feito durante a seleção e/ou

10 Erro durante o processo (um código de erro de identificação e uma descrição são exibidos).

Pré-Renovação de Portfólio (como parte da etapa: Portfólio)

15 A pré-renovação de portfólio é lançada um mês ou dois meses antes da renovação de portfólio para os produtos que requerem a mesma. O processo de pré-renovação é lançado dois meses antes do vencimento de apólice que cobre o seguinte:

A mesma atualiza os valores de apólice e avalia os prêmios para o próximo período e/ou

20 Bônus/Penalidade, Soma segurada, Atualizações Automatizadas e Outros Processos Automatizados (descontos e cobranças extras) e/ou

A mesma processa as notas de vencimento, mas, não emite o movimento e/ou

25 Este processo congela as apólices durante sua execução (apenas as cotações de renovação podem ser feitas nesta apólice, entretanto, as apólices pode ser manualmente descongeladas por um usuário autorizado).

30 Como um exemplo, a empresa envia informação sobre a renovação de apólice para o cliente, a fim de informá-lo das novas condições de renovação ou do cancelamento do contrato (devido a emissões de reivindicações). Posteriormente, as cotações baseadas na pre-renovação podem ser feitas.

O método acima também pode compreender uma etapa relacionada a comissões.

Etapa: Comissões

As comissões podem ser designadas para realizar uma ou mais das seguintes etapas e/ou compreender um ou mais dos seguintes recursos/efeitos:

- 5 Recuperar os valores de tabela a partir do modulo de canais de Distribuição (produto – classe prêmio, grupo de afinidade, agente, etc.).
Manualmente dentro de limites pré-definidos.
Distribuição de comissionamento entre diferentes agentes dentro de limites pré-definidos.
- 10 Diferentes níveis de comissão podem ser configurados dependendo
das primeiras anuidades e sucessivas e/ou
da função do agente, produtor ou coletor.
Adicionalmente, o método acima pode permitir e/ou compreender
- 15 uma ou mais:
Uma tela de resultado de classificação como uma GUI específica pode ser proporcionada, que permite a exibição das quantias de prêmio, descontos, sobretaxas e/ou taxas na apólice, objeto e/ou nível de classe prêmio.
- 20 O período de prestação e os sucessivos são determinados, de acordo com ambas as formas de pagamento, e a informação proporcionada pela Fábrica de Produto
Os detalhes dos prêmios resultantes são disponíveis através do objeto ou classe prêmio.
- 25 No caso de classificação manual ou semi-automática de um produto, isto pode ser feito nas telas de administração de Apólice.

Programas principais

As seguintes etapas do processo de emissão apólice são resumidas:

30 De apólice completa à incompleta

Os movimentos de Novo Negócio, endossos e restabelecimentos são feitos pelo programa KCILMS01. No caso de cancelamentos, os mes-

mos são feitos pelo programa KCILMM03.

Entrada de dados

A entrada de dados é realizada pelos programas para obter informação da apólice em um nível de apólice (KCILMQ03), objeto (KCILMB03) e classe prêmio (KCILMD03). Os programas para modificar esta informação são KCILMQ02 (dados de apólice geral), KCILMQ04 (dados adicionais), KCILMB04 (objeto) e KCILMD04 (classe prêmio). Estes membros usam o programa KCILMZ06 para chamar a Fábrica de Produto executável para validar todos os dados com as especificações de produto, para introduzir o gerenciamento de competência, etc.

Validação/Classificação

Uma vez que a validação é feita em cada nível (apólice, objeto, classe prêmio), o sistema chama novamente o executável para fazer uma validação geral em todos os níveis (isto é feito pelo programa KCILMZ06 novamente). Após as validações de dados o produto executável é usado novamente para classificar (obter o prêmio). Finalmente, existe uma chamada do Programa de Prestação (KCILMZ00), para dividir o prêmio em cada um dos recibos resultantes, de acordo com ambas as formas de pagamento selecionadas no prêmio anual de um movimento e na informação proporcionada pela Fábrica de Produto.

Emissão de Apólice

O processo de emissão valida e consolida todos os dados introduzidos que alteram o estado da apólice de incompleta para completa. O processo é iniciado pelo programa KCILMQ05 que chama dinamicamente diversos subprogramas que dependem do produto, do movimento e de outros parâmetros.

Uma ou mais das seguintes tarefas pode ser compreendida no processo de emissão:

Validação dos parágrafos e/ou emissão de recibo e/ou resseguro, etc. com a Fábrica de Produto.

Realizar operações adicionais (calcular comissões e/ou composição cláusula, etc.).

Consolidar a apólice com todas as quitações. Este processo é feito pelo KCILMZ05.

Proporcionar/gerar interfaces de resseguro e/ou cosseguro.

Processo Econômico Relacionado, tal como, emissão de recibo.

5 De maneira vantajosa, a Fábrica de Produto é completamente integrada com a administração de Apólice, de modo que alterações feitas no criador dos produtos sejam prontamente disponíveis para os usuários. More especificamente,

10 a estrutura de produto é definida na Fábrica de Produto. O pedido de administração de apólice é capaz de interpretar o mesmo e adaptar as telas, tais como, a GUI mostrada nas figuras 28 e 29 e os bancos de dados, a fim de armazenar a informação requerida, sem a necessidade de um conhecimento destas;

15 as validações de produto e a ação a ser tomada quando ocorrem incidentes são definidas na Fábrica de Produto e a administração de apólice apresenta a mensagem de erro apropriada que pode ser definida na Fábrica de Produto para o usuário;

20 os cálculos podem ser definidos na Fábrica de Produto e o dado financeiro é salvo em um banco de dados de informação econômica, a partir do qual a administração de apólice mostra os resultados detalhados para o usuário.

25 A figura 31 é uma visão geral esquemática de como a Fábrica de Produto e a administração de apólice são integradas uma à outra. Conforme pode ser visto, a estrutura de um contrato/apólice é diretamente ligada à estrutura de produto, conforme definido na Fábrica de Produto. Um contrato pode ser um grupo de apólices que pode ser agrupado em diferentes níveis. Principalmente, estes agrupamentos podem ser usados para agrupar a emissão de recibos e gerar relatórios e estatísticas agregadas. Os agrupamentos de divisão e subdivisão nas políticas de contrato são opcionais. As
30 apólices de um contrato podem ser diretamente agrupadas com o contrato.

Os objetos de risco disponíveis e as garantias (Classe prêmio na terminologia presente) são aqueles definidos na Fábrica de Produto sob o

produto que a apólice se baseia. Em outras palavras, o método descrito acima, permite de maneira vantajosa o gerenciamento de todas as aquisições de novo negócio e necessidades de manutenção de portfólio de apólice de uma empresa de seguro, de acordo com as regras predefinidas na Fábrica

5 de Produto

Em particular, as transações de negócio mais comuns podem ser incluídas:

- Cotações e/ou
- Novo negócio e/ou
- 10 Endossos e/ou
- Renovações e/ou
- Cancelamento e/ou
- Restabelecimentos e/ou
- Declarações e/ou
- 15 Saneamento de Apólice e/ou
- Gerenciamento de Competência e/ou
- Outros...

A figura 32 mostra um modelo conceitual de uma visão geral de arquitetura técnica 200 para implementar o método descrito acima. Conforme

20 mostrado, a arquitetura de usuário 202 proporciona usabilidade e interação. A arquitetura de aplicações 204 representa uma visão integrada dos sistemas para solucionar problemas de negócios. A arquitetura de desenvolvimento 206 proporciona/define um ou mais dentre o seguinte:

- Metodologia
- 25 Padrões
- Procedimentos
- Ferramentas.

A arquitetura de operações 208 proporciona/define um ou mais dentre o seguinte:

- 30 Suporte de Operações
- Segurança
- Disponibilidade.

A arquitetura de dados 210 proporciona/define um ou mais dentre o seguinte:

Definição

Visões lógicas e físicas

5 Administração.

A arquitetura de execução 212 proporciona/define um conjunto de módulos e/ou programas e/ou objetos e/ou APIs que podem isolar os aspectos técnicos.

10 A arquitetura de impressão 214 proporciona/define um conjunto de ferramentas e/ou procedimentos para gerenciar e imprimir facilidades.

A seguir, como uma visão geral, a estrutura repositória e estrutura de tabela serão descritas, à medida que podem ser aplicadas de acordo com o método/sistema descrito neste pedido.

15 De acordo com o presente pedido, na Administração de Apólice, os valores são dinamicamente armazenados, o que significa que dependem de como os elementos foram definidos no produto, eles podem ser armazenados (dentro do mesmo banco de dados) em um campo ou em outro.

20 Isto retrata o banco de dados de Administração de Apólice com a flexibilidade para ELEMENTOS ADICIONAIS (aqueles definidos no Nível de Apólice, Nível de Objeto e Nível de Cobertura), porque o mesmo permite o aumento ou redução do número de elementos adicionais a ser implementado em cada cliente sem afetar o banco de dados. Portanto, uma vez que a estrutura repositória é definida, os elementos adicionais podem ser incluídos no repositório sem alterar a estrutura do repositório.

25 O seguinte exemplo tem para o propósito de ilustrar como os valores lançados para os elementos adicionais (no nível de apólice, nível de objeto e/ou nível de cobertura) são armazenados no banco de dados de Administração de Apólice:

30 Uma vez que o produto, isto é, o conjunto de arquivos executáveis interrelacionados é gerado, na tabela KTPA82T todas as designações de elemento são armazenadas, isto é, a KTPA82T define/incorpora o local físico exato em que o valor dos elementos será armazenado dentro das tabe-

las de Administração de Apólice. Nesta tabela TODOS os vetores relacionados a TODOS os níveis são armazenados. Portanto, antes de recuperar qualquer valor para um certo elemento, é obrigatório acessar esta tabela a fim de conhecer o local físico em que os valores se situam. No(s) exemplo(s)

5 a seguir para cada nível são descritos:

ELEMENTO DE APÓLICE ADICIONAL:

KTPA82T: O banco de dados com a designação de elemento. O exemplo a seguir mostra como o dito banco de dados pode ser acessado para recuperar o vetor para um elemento no nível de apólice. No caso, o valor para um elemento exemplificativo CDPOSTCO (código postal) deve ser recuperado. O banco de dados pode compreender a seguinte entrada:

<u>CDPRODTE</u>	<u>CDPRODCO</u>	<u>CDOBJTP</u>	<u>CDMCT</u>	<u>CDELEME</u>	<u>NOCAMFI</u>	<u>NURELREC</u>
MOTOR	MOTOR_1	000000000	00000000	CDPOSTCO	TXCAOP1	1

Então, com NOCAMFI=TXCAOP1 e NURELREC=1, as seguintes tabelas são acessadas:

- KCIM52T (Incompleta)
- KCIM16T (Completa).

Estas tabelas KCIM52T e KCIM16T são exatamente as mesmas e dependem se uma apólice emitida (complete) ou uma apólice não emitida, que está pendente (incompleta) é trabalhada. Portanto, KCIM16T ou KCIM52T são acessadas, respectivamente, porém, as tabelas serão acessadas do mesmo modo, à medida que ambas são tabelas idênticas.

A fim de acessar e recuperar o valor, a informação é recuperada a partir dos seguintes campos (NURELREC=1 significa que para uma faixa de registros encontrada, apenas a primeira será selecionada, isto é, apenas a primeira linha será usada):

<u>CDNUMPOL</u>	<u>CTVRSPOL</u>	<u>TXCAOP1</u>	<u>TXCAOP2</u>
99788888801	1	VALOR 1	0000000
99788888801	1	VALOR 2	0000000
99788888801	2	VALOR 3	0000000

Então, "VALOR 1" é o valor específico para o elemento CDPSTCO.

Usando a terminologia do exemplo acima, a estrutura repositória pode compreender o banco de dados KTPA82T. Os elementos repositórios podem ser as linhas definidas acima:

	<u>CDPRODTE</u>	<u>CDPRODCO</u>	<u>CDOBJTP</u>	<u>CDMCT</u>	<u>CDELEME</u>	<u>NOCAMPFI</u>	<u>NURELREC</u>
5	MOTOR	MOTOR_1	000000000	00000000	CDPOSTCO	TXCAOP1	1

As tabelas KCIM52T e KCIM16T podem ser compreendidas pela estrutura de tabela.

ELEMENTO DE OBJETO ADICIONAL:

KTPA82T: O banco de dados com a designação de elemento. O exemplo a seguir mostra como o dito banco de dados pode ser acessado para recuperar o vetor para um elemento no nível de apólice. No caso, o valor para um elemento exemplificativo CDLOCAT (localização) deve ser recuperado. O banco de dados pode compreender a seguinte entrada:

	<u>CDPRODTE</u>	<u>CDPRODCO</u>	<u>CDOBJTP</u>	<u>CDMCT</u>	<u>CDELEME</u>	<u>NOCAMPFI</u>	<u>NURELREC</u>
15	MOTOR	MOTOR_1	000005000	00000000	CDLOCAT	IMCAOP1	3

Então, com NOCAMFI=IMCAOP1 e NURELREC=3, as seguintes tabelas são acessadas:

KCIM54T (Incompleta)

KCIM18T (Completa).

Estas tabelas KCIM54T e KCIM18T são exatamente as mesmas e dependem se uma apólice emitida (complete) ou uma apólice não emitida, que está pendente (incompleta) é trabalhada. Portanto, KCIM18T ou KCIM54T são acessadas, respectivamente, porém, as tabelas serão acessadas do mesmo modo, à medida que ambas são tabelas idênticas.

A fim de acessar e recuperar o valor, a informação é recuperada a partir dos seguintes campos (NURELREC=3 significa que para uma faixa de registros encontrada, apenas a primeira será selecionada, isto é, apenas a terceira linha será usada):

	<u>CDNUMPOL</u>	<u>CTVRSPOL</u>		<u>IMCAOP1</u>	<u>IMCAOP2</u>
30	99788888801	1		VALOR 1	0000000
	99788888801	1		VALOR 2	0000000
	99788888801	2		VALOR 3	0000000

Então, o "VALOR 3" é o valor específico para o elemento C-DLOCAT.

ELEMENTO DE COBERTURA ADICIONAL:

- 5 KTPA82T: O banco de dados com a designação de elemento. O exemplo a seguir mostra como o banco de dados pode ser acessado para recuperar o vetor para um elemento no nível de apólice. No caso, o valor para o elemento IMLIMIT (Quantia Limite) deve ser recuperado. O banco de dados pode compreender a seguinte entrada:

	<u>CDPRODTE</u>	<u>CDPRODCO</u>	<u>CDOBJTP</u>	<u>CDMCT</u>	<u>CDELEME</u>	<u>NOCAMPFI</u>	<u>NURELREC</u>
10	MOTOR	MOTOR_1	000005000	00000000	IMLIMIT	IMCAOP5	2

Então, com NOCAMFI=IMCAOP5 e NURELREC=2, as seguintes tabelas são acessadas:

KCIM53T (Incompleta)

KCIM23T (Completa).

- 15 Estas tabelas KCIM53T e KCIM18T são exatamente as mesmas e dependem se uma apólice emitida (complete) ou uma apólice não emitida, que está pendente (incompleta) é trabalhada. Portanto, KCIM23T ou KCIM53T são direcionadas, respectivamente, porém, as tabelas serão acessadas do mesmo modo, à medida que ambas são tabelas idênticas.
- 20 modo, à medida que ambas são tabelas idênticas.

A fim de acessar e recuperar o valor, a informação é recuperada a partir dos seguintes campos (NURELREC=2 significa que para uma faixa de registros encontrada, apenas a primeira será selecionada, isto é, apenas a segunda linha será usada):

25	<u>CDNUMPOL</u>	<u>CTVRSPOL...</u>	<u>IMCAOP1</u>	<u>IMCAOP2</u>	<u>IMCAOP3</u>	<u>IMCAOP4</u>	<u>IMCAOP5</u>
	99788888801	1	0000000	0000000			VALOR 1
	99788888801	1	0000000	0000000			VALOR 2
	99788888801	2	0000000	0000000			VALOR 3

Então, "VALOR 2" é o valor específico para o elemento IMLIMIT.

- 30 Uma visão geral esquemática dos processos e inter-relações, para gerar documentos e para distribuir impressão, é mostrada na figura 33.

Dado o que foi dito acima, os seguintes princípios de projeto po-

dem ser aplicados e realizados:

- Arquitetura de múltiplos canais em 3 filas.
 - Arquitetura "baseada em "Componente": aplicação de princípios de integração de aplicação de empreendimento para inter-componentes.
- 5 • Portabilidade: a camada de serviços protege a aplicação a partir da implementação de software subjacente.
- Múltiplas linguagens (Interface de usuário e toda saída de documento o), múltiplas moedas, múltiplas empresas.
 - Geração de Documento Central & arquitetura de publicação flexível
- 10 (centralmente & distribuída, on-line & em lote).

Uma visão geral exemplificativa de um possível sistema em 3 filas é mostrada na figura 34, que mostra os elementos específicos da arquitetura em 3 filas e sua conexão.

- 15 A Tabela 1 proporciona de maneira exemplificativa uma visão geral de uma implementação específica que lista os componentes usados. Outra configuração técnica que pode ser possível, pode ser baseada em diferentes plataformas. Os componentes opcionais podem ser substituídos pelos componentes e/ou interfaces de aplicação existentes. Na tabela 1, são usados os termos que podem ser marcas registradas de terceiros.

Client	• Internet: HTML, Javascript (Thin Client) • Intranet: Java (Fat Client with automatic pull update system)
WebServer	• Apache webserver (for webpage publishing) on winNT – Current configuration at the "Mexican" solution • Located in demilitarized zone between firewalls
Security & Access Mgmt	• Standard SSL over VPN network
Application Server	• Websphere on WinNT (compatible with standard J2EE) • JAVA, EJB, JSP, SERVLET
Middleware	• CICS Transaction Server (IMS scenario to be analysed)
Back-end server	• Highest range OS/390 • Languages: Cobol, DB2
Document Management & Imaging (optional)	• FileNet for imaging and storage • DocOne: document generation
Datawarehouse (optional)	• Data warehouse: IBM AIX, DB2 UDB 6.0
Accounting (optional)	• SAP

Tabela 1

Na figura 35, mostra-se uma visão geral 300 de uma arquitetura técnica que compreende particularmente a relação de componentes específicos da arquitetura. Como um exemplo, um cliente HTML 302 e um cliente Java 304 são proporcionados. O cliente HTML 302 compreende uma estação de trabalho de cliente 306. O cliente Java 304 compreende uma estação de trabalho de cliente 308. A estação de trabalho de cliente 306 do cliente HTML 304 pode ser conectada a um servidor web 310 e/ou um servidor de aplicação 312. O servidor web 310 pode ser adaptado para chamadas de serviço de anfitrião e/ou legado. Além disso, o servidor web 310 pode compreender lógica de edição básica e/ou lógica de navegação. O servidor web 310 também pode ser adaptado para gerar apresentações. Se aplicável, o mesmo também pode requerer o servidor de aplicação 312. Além disso, o servidor de aplicação 312 pode ser adaptado para receber e/ou armazenar e/ou gerar dados de referência duplicados. Se aplicável, este também pode requerer o servidor web 310. Este pode ser particularmente o caso, se o servidor web 310 e o servidor de aplicação 312 forem combinados em uma entidade.

O servidor de aplicação 312 é conectado a um servidor de integração 314. O servidor de integração 314 pode compreender lógica de integração. Se aplicável, o servidor web 312 também pode ser conectado ao servidor de integração 314. O servidor de integração pode ser conectado a um ou sistemas de legado 316. Cada sistema de legado 316 pode compreender lógica de negócio de legado.

A estação de trabalho de cliente 308 do cliente Java 304 é conectada a um anfitrião 318. O anfitrião pode compreender um ou mais dos seguintes componentes de lógica:

- Lógica de edição
- Lógica de negócio
- Lógica de acesso de dados
- Lógica de interface
- Lógica de lote.

Além disso, o anfitrião 318 pode receber e/ou armazenar e/ou

gerar dados de transação e/ou dados de referência.

A seguir, os elementos, conforme mostrado na figura 35, são descritos em detalhes adicionais:

ANFITRIÃO: Este representa o computador central. No anfitrião,
5 todo o banco de dados pode ser definido e todas as lógicas de negócio implementadas em programas cobol podem permanecer. Estes programas podem acessar os bancos de dados para atualizar e/ou manter os dados que também podem ser usados para fazer cálculo, etc.

SERVIDOR DE APLICAÇÃO (janelas, UNIX...): Neste ambiente
10 uma lógica de Cliente Magro pode ser executada. Em outras palavras, o servidor de aplicação define a localização para manter a ordem na qual as páginas da web devem ser lançadas. Esta aplicação pode ajustar a ordem na qual as telas aparecem ou ainda a necessidade de aparecerem.

SERVIDOR WEB: É o servidor que permite o fornecimento de
15 páginas físicas para o servidor de aplicação. O servidor de aplicação (baseado na ordem) pode solicitar as páginas para o servidor web que é o provedor destas páginas.

ESTAÇÃO DE TRABALHO DE CLIENTE: Isto é, o PC a partir do
20 qual o usuário pode acessar o sistema, de acordo com o presente pedido. O mesmo pode ser o terminal a partir do qual alguém deseja se registrar no dito sistema, por exemplo, para executar um ou mais arquivos executáveis a partir do conjunto de arquivos executáveis através da GUI descrita acima.

SERVIDOR DE INTEGRAÇÃO & SERVIDOR DE LEGADO: Em
25 alguns casos alguns clientes podem desejar manter seus dados de legado não integrados com o sistema acima, de acordo com o presente pedido, mas, armazenados em um servidor diferente que é chamado de SISTEMAS DE LEGADO. Então, através de um servidor de integração os dados armazenados no sistema de legado podem ser recuperados e enviados para o servidor de aplicação, quando necessário.

30 A figura 36 proporciona uma visão geral da arquitetura técnica para a geração de interface automática. A arquitetura de execução mostrada proporciona um mecanismo de geração automática para muitos, particular-

mente, todos os componentes que interferem no início de comunicação a partir de uma IDL (Linguagem de Definição de Interface). De maneira vantajosa, esta geração automática permite que as aplicações tenham que lidar com a definição das mensagens de entrada e/ou saída em um dicionário de dados, porém, não com a sua geração física (em forma de código).

A seguir, conforme mostrado na figura 36, os elementos são descritos em detalhes adicionais. Em particular, a figura 36 mostra como a EXTREMIDADE ANTERIOR comunica DADOS para a EXTREMIDADE POSTERIOR. Em outras palavras, e usando um exemplo, a extremidade anterior sob a forma de uma estação de trabalho pode ser um banco. Um usuário da estação de trabalho pode receber e lançar uma demanda/solicitação de empréstimo. A extremidade posterior pode ser representada pelas operações que são feitas, por exemplo, no departamento administrativo, a fim de determine se o banco aprova o empréstimo ou não. Então, de acordo com a visão geral esquemática mostrada na figura 36, os termos usados podem ser definidos da seguinte maneira:

ESTAÇÃO DE TRABALHO: O PC a partir do qual o usuário pode acessar as telas, isto é, a GUI/GUIs, conforme descrito neste pedido. O usuário pode usar a(s) tela(s) AIS para enviar e processar dados (situados nos campos) ao longo das classes.

CLASSE EXTERNA: Uma faixa de campos (dados) que são enviados para o MIDDLEWARE para serem processados pelos programas.

CLASSE INTERNA: Uma faixa de campos (dados) que são recebidos pelo MIDDLEWARE (geralmente cálculos ou resultados de acesso aos bancos de dados) que são previamente enviados pelos arquivos executáveis, que podem ser programas cobol a serem exibidos nas telas.

STUB: Ordens a partir das quais os campos são empacotados para serem enviados.

MIDDELEWARE: Software que comunica os dados enviados através das telas, isto é, a GUI para a extremidade posterior, em outras palavras, o software que comunica os dados lançados nas telas com o programas situados no computador central. O mais usado é a CTG (PORTA DE

TRANSAÇÃO CICS).

OS390: O computador central ou o anfitrião onde todos os programas e bancos de dados são situados.

5 BANCO DE DADOS: A estrutura de dados onde todas as informações relacionadas às telas e/ou resultados de cálculos são armazenadas.

ROTINA DE ACESSO DE DADOS: Os arquivos executáveis, tais como, programas COBOL onde permanece toda a lógica de negócio sobre a aplicação.

10 CÓPIA EXTERNA: Similar, ainda exatamente as mesmas que as Classes Java, porém, para COBOL. Em outras palavras, esta pode ser a faixa de campos que são enviados para o middleware a fim de serem enviados para a tela e/ou podem ser o resultado de algum cálculo e/ou acesso de programa.

15 CÓPIA INTERNA: O mesmo que "CÓPIA EXTERNA", porém, ao inverso. Isto representa a faixa de dados que é recebida para os programas cobol a fim de realizar algum cálculo e/ou acesso de programa.

ESQUEMA: Similar, ainda o mesmo que STUB, que é a ordem a partir da qual os campos são empacotados para serem enviados.

20 A fim de verificar que o processo/método descrito acima pode ser realizado de maneira apropriada, a definição das CLASSES e dos COPYBOOKS deve combinar, em particular, deve combinar exatamente uma com a outra. Portanto, proporciona-se um dicionário de dados, que é um banco de dados onde todos os campos são definidos (tais como, comprimento, formato, etc...). A partir deste ponto existe um processo chamado
25 IDL que gera automaticamente em formato e comprimento idênticos de Classes Java e Cópias Cobol, a fim de garantir a coerência. De maneira vantajosa, ao usar isto, o programador não precisa gerar manualmente os ditos formatos idênticos.

30 Geralmente, quando o método descrito acima é executado, à medida que o nome de usuário e a senha de serviço de segurança são autenticados, permitindo o acesso ao sistema. O sistema pode recuperar o perfil de segurança do usuário, que define quais funções o usuário pode aces-

sar. As funções podem definir o acesso necessário para abordar a funcionalidade de negócio, tais como

acesso aos cenários: áreas de procedimentos de negócio e/ou

controle de acesso: botões de acesso para as janelas ou servi-

5 ços específicos e/ou

acesso aos recursos lógicos: para desativar /ativar o acesso às tabelas específicas

Acessos (consulta & modificação) e parâmetros de competência são ajustados seguindo as estruturas de negócio (papéis e localização).

10 Além do que foi dito acima, a geração automática de diferentes pacotes de aplicação em diversas linguagens fora de uma única versão do código de aplicação pode ser proporcionada.

Referindo-se à figura 37, um sistema exemplificativo para implementar a invenção inclui um dispositivo de computação de propósito geral sob a forma de um ambiente de computação convencional 20 (por exemplo, 15 computador pessoal), que inclui uma unidade de processamento 22, uma memória de sistema 24 e um barramento de sistema 26 que acopla vários componentes de sistema incluindo a memória de sistema 24 na unidade de processamento 22. A unidade de processamento 22 pode realizar operações 20 aritméticas, lógicas e/ou de controle ao acessar a memória de sistema 24. A memória de sistema 24 pode armazenar informações e/ou instruções para uso em combinação com a unidade de processamento 22. A memória de sistema 24 pode incluir memória volátil e não volátil, tais como, memória de acesso aleatório (RAM) 28 e memória somente leitura (ROM) 30. Um siste- 25 ma de entrada/saída básico (BIOS) que contém as rotinas básicas que ajudam a transferir informações entre os elementos dentro do computador pessoal 20, tal como, durante a inicialização, pode ser armazenado na ROM 30. O barramento de sistema 26 pode ser qualquer um entre diversos tipos de estruturas de barramento que incluem um barramento de memória ou contro- 30 lador de memória, um barramento periférico e um barramento local que usa qualquer uma entre uma variedade de arquiteturas de barramento.

O computador pessoal 20 pode incluir adicionalmente um drive

de disco rígido 32 para ler e escrever em um disco rígido (não mostrado), e um drive de disco externo 34 para ler ou escrever em um disco removível 36. O disco removível pode ser um disco magnético para um driver de disco magnético ou um disco óptico, tal como um CD ROM para um drive de disco óptico. O drive de disco rígido 34 e o drive de disco externo 34 são conectados ao barramento de sistema 26 através de uma interface de drive de disco rígido 38 e uma interface de drive de disco externo 40, respectivamente. Os drives e seus meios legíveis por computador associados proporcionam armazenamento não volátil de instruções legíveis por computador, estruturas de dados, módulos de programa e outros dados para o computador pessoal 20. As estruturas de dados podem incluir dados relevantes da implementação do método, conforme descrito em mais detalhes acima. O dado relevante pode ser organizado em um banco de dados, por exemplo, um banco de dados relacional ou de objeto.

Embora o ambiente exemplificativo descrito no presente documento empregue um disco rígido (não mostrado) e um disco externo, deve ser avaliado por aqueles versados na técnica que outros tipos de meios legíveis por computador que podem armazenar dados que são acessíveis por um computador, tais como, fitas magnéticas, cartões de memória flash, discos de vídeo digital, memórias de acesso aleatório, memória somente leitura, e similares, também podem ser usados no ambiente de operação exemplificativa.

Inúmeros módulos de programa podem ser armazenados no disco rígido, disco externo, ROM 30 ou RAM 28, incluindo um sistema de operação (não mostrado), um ou mais programas de aplicação 44, outros módulos de programa (não mostrados) e dados de programa 46. Os programas de aplicação podem incluir pelo menos uma parte da funcionalidade, conforme detalhado nas figuras 1 a 36.

Um usuário pode lançar comandos e informações, conforme discutido abaixo, no computador pessoal 20 através de dispositivos de entrada, tais como, um teclado 48 e mouse 50. Outros dispositivos de entrada (não mostrados) podem incluir um microfone (ou outros sensores), vídeo game,

controle de jogo, digitalizador, ou similares. Estes e outros dispositivos de entrada podem ser conectados à unidade de processamento 22 através de uma interface de porta em série 52 que é acoplada ao barramento de sistema 26, ou podem ser coletados por outras interfaces, tal como, uma interface de porta paralela 54, porta de jogo ou um barramento serial universal (USB). Ademais, as informações podem ser impressas usando a impressora 56. A impressora 56 e outros dispositivos de entrada/saída paralelos podem ser conectados à unidade de processamento 22 através da interface de porta paralela 54. Um monitor 58 ou outro tipo de dispositivo de tela também é conectado ao barramento de sistema 26 através de uma interface, tal como uma entrada/saída de vídeo 60. Além do monitor, o ambiente de computação 20 pode incluir outros dispositivos de saída periféricos (não-mostrados), tais como, altofalantes ou outra saída audível.

O ambiente de computação 20 pode se comunicar com outros dispositivos eletrônicos, tal como, um computador, telefone (com fio ou sem fio), assistente digital pessoal, televisão, ou similares. Para se comunicar, o computador/ambiente de computação 20 pode operar em um ambiente de rede que usa conexões em ou mais dispositivos eletrônicos. A figura 37 mostra o ambiente de computador em rede com o computador remoto 62. O computador remoto 62 pode ser outro ambiente de computação, tal como, um servidor, roteador; uma rede PC, um dispositivo par ou outro nó de rede comum, e pode incluir muitos ou todos os elementos descritos acima relativos ao ambiente de computação 20. As conexões lógicas mostradas na figura 37 incluem uma rede de área local (LAN) 64 e uma rede de área ampla (WAN) 66. Tais ambientes de rede são comuns em escritórios, redes de computador empresariais-amplos, intranets e a Internet.

Quando usado em um ambiente de rede LAN, o ambiente de computação 20 pode ser conectado à LAN 64 através de uma rede I/O 68. Quando usado em um ambiente de rede WAN, o ambiente de computação 20 pode incluir um modem 70 ou outros meios para estabelecer as comunicações através da WAN 66. O modem 70, que pode ser interno ou externo ao ambiente de computação 20, é conectado ao barramento de sistema 26

através da interface de porta em série 52. Em um ambiente em rede, os módulos de programa mostrados em relação ao ambiente de computação 20 ou porções destes podem ser armazenados em um dispositivo de armazenamento de memória remoto residente ou acessível ao computador remoto 62.

- 5 Além disso, outros dados relevantes para o método implementado por computador para gerar arquivos de computador executáveis interrelacionados que podem ser produtos de seguro ou relacionados a produtos de seguro e/ou relacionados às aplicações de gerenciamento de reivindicação de seguro (descritas em mais detalhes acima) e/ou relacionados à geração de documentos de seguro, e/ou relacionados à geração de produtos adicionais, 10 tais como, produtos de imóveis, produtos de serviços financeiros e/ou documentos, podem ser residente ou acessíveis através do computador remoto 62. Os dados podem ser armazenados, por exemplo, em um objeto ou um banco de dados. Será avaliado que as conexões de rede mostradas são exemplificativas e outros meios de estabelecer uma conexão de comunicações 15 entre os dispositivos eletrônicos podem ser usados.

O sistema de computador descrito acima é apenas um exemplo do tipo de sistema de computação que pode ser usado para implementar o método implementado por computador descrito acima.

20

Lista de Referências Numéricas

	1	GUI
	2a	campo
	2b	campo
	4a	campo
25	4b	campo
	6a	campo
	6b	campo
	8	campo
	10	janela
30	12	caixa
	16	botão
	18	botão

	20	ambiente de computação convencional
	22	unidade de processamento
	24	memória de sistema
	26	barramento de sistema
5	28	memória de acesso aleatório (RAM)
	30	memória somente leitura (ROM)
	32	drive de disco rígido
	34	drive de disco externo
	36	disco removível
10	38	interface de drive de disco rígido
	40	interface de drive de disco externo
	44	um ou mais programas de aplicação
	46	dados de programa
	48	teclado
15	50	mouse
	52	interface de porta serial
	54	interface de porta paralela
	56	impressora
	58	monitor
20	60	entrada/saída de vídeo
	62	computador remoto
	64	rede de área local (LAN)
	66	rede de área ampla (WAN)
	68	rede I/O
25	70	um modem
	101	GUI
	110	listas suspensas
	112	botões
	114	campos de entrada
30	114a	campo de entrada
	116	janela de entrada/saída
	118a - 118i	elementos de exibição

	118'a - 118'b	elementos de exibição
	120a - 120i	campos de entrada
	120'a - 120'b	campos de entrada
	122	barra de rolagem
5	124	cabeçalho
	126	cabeçalho
	200	visão geral de arquitetura técnica
	202	arquitetura de usuário
	204	arquitetura de aplicações
10	206	arquitetura de desenvolvimento
	208	arquitetura de operações
	210	arquitetura de dados
	212	arquitetura de execução
	214	arquitetura de impressão
15	300	visão geral
	302	cliente HTML
	304	cliente Java
	306	estação de trabalho de cliente
	308	estação de trabalho de cliente
20	310	servidor web
	312	servidor de aplicação
	314	servidor de integração
	316	sistemas de legado
	318	anfitrião

REIVINDICAÇÕES

1. Método implementado por computador para adaptar e utilizar um ambiente de computação, em que o ambiente de computação compreende

5 um conjunto de arquivos executáveis,
 uma estrutura repositória,
 uma estrutura de tabela, em que
 a estrutura repositória e a estrutura de tabela são associadas
uma à outra,

10 e
 interface gráfica de usuário (GUI 101) que é adaptada para iniciar
 a execução dos arquivos executáveis,
 a leitura de dados e/ou escrita de dados na estrutura de tabela, e
 fornecimento de dado de saída como texto legível por humano,
15 o método implementado por computador que tem as etapas de:
 proporcionar uma definição de uma alteração do ambiente de
computação em um lado de cliente,

 adaptar os arquivos executáveis, de acordo com a definição da
alteração do ambiente de computação em um lado de servidor,

20 introduzir pelo menos um elemento repositório adicional na estrutura repositória no lado de servidor, em que pelo menos um elemento repositório adicional corresponde à adaptação dos arquivos executáveis,

 adaptar a estrutura de tabela no lado de servidor, que corresponde ao elemento repositório adicional da estrutura repositória, e

25 acessar a estrutura de tabela adaptada no lado de servidor através da dita GUI (101) e/ou executar um ou mais arquivos executáveis no lado de servidor através da dita GUI (101), enquanto mantém substancialmente o esboço visual da dita GUI (101).

2. Método implementado por computador, de acordo com a reivindicação 1, em que o arquivos executáveis e a dita GUI (101) acessam a mesma estrutura de tabela, em que o acesso compreende escrever dados e/ou ler dados a partir da mesma estrutura de tabela.

3. Método implementado por computador, de acordo com a reivindicação 1 ou 2, em que a adaptação do ambiente de computação é substancialmente livre de cópia da estrutura repositória e/ou da estrutura de tabela.

4. Método implementado por computador, de acordo com pelo menos uma das reivindicações precedentes, em que quando adapta o ambiente de computação, a estrutura repositória é substancialmente mantida.

5. Método implementado por computador, de acordo com pelo menos uma das reivindicações precedentes, em que

usar a dita GUI (101), o dado de saída é gerado pelo ambiente de computação,

e em que

após a adaptação do ambiente de computação, que usa a dita GUI (101), o dado de saída adicional é gerado pelo dito ambiente de computação adaptado.

6. Método implementado por computador, de acordo com pelo menos uma das reivindicações precedentes, que compreende adicionalmente a etapa de adaptar a dita GUI (101), de acordo com pelo menos um elemento repositório adicional na estrutura repositória ao introduzir pelo menos um elemento de exibição (118a a 118i) em uma entidade de entrada/saída dinâmica (116) da dita GUI (101), enquanto mantém as entidades de entrada/saída restantes (110, 112, 114) da dita GUI (101) substancialmente inalteradas.

7. Método implementado por computador, de acordo com a reivindicação 6, em que ao introduzir pelo menos um elemento adicional de exibição (118a a 118i) na dita entidade de entrada/saída dinâmica (116) da dita GUI (101), o tamanho da dita entidade de entrada/saída dinâmica (116) é mantido

8. Método implementado por computador, de acordo com a reivindicação 7, em que a dita entidade de entrada/saída dinâmica (116) da GUI (101) é implementada como uma lista suspensa, se o tamanho da dita entidade de entrada/saída dinâmica (116) for menor que o tamanho comum de todos os elementos de exibição (118a a 118i) da dita entidade de entrada/saída dinâmica (116).

da/saída (116).

9. Método implementado por computador, de acordo com pelo menos uma das reivindicações 6 a 8, em que

5 Para exibir um elemento de exibição (118a a 118i) na entidade de entrada/saída dinâmica (116) o conteúdo do dito elemento de exibição (118a a 118i) é lido a partir de um elemento de tabela associado

e/ou

10 para armazenar o conteúdo (120a a 120i) de um elemento de exibição (118a a 118i) da entidade de entrada/saída dinâmica (116), o conteúdo (120a a 120i) do dito elemento de exibição (118a a 118i) é escrito em um elemento de tabela associado.

10. Método implementado por computador, de acordo com pelo menos uma das reivindicações 6 a 9, em que

15 os elementos de tabela armazenados na dita estrutura de tabela são armazenados em uma ordem de tabela predeterminada,

em que

elementos de exibição (118a a 118i) exibidos na entidade de entrada/saída dinâmica (116) são exibidos em uma ordem de exibição predeterminada,

20 e em que

a ordem de tabela predeterminada é diferente da ordem de exibição predeterminada.

11. Método implementado por computador, de acordo com a reivindicação 10, em que um dispositivo de ordem de elemento é usado

25 para ler o conteúdo de um elemento de tabela no elemento de exibição associado (118a a 118i) na ordem de exibição,

e

para escrever o conteúdo (120a a 120i) de um elemento de exibição (118a a 118i) no elemento de tabela associado na ordem de tabela.

30 12. Produto de programa de computador, que compreende instruções legíveis por computador, que quando carregadas na memória de um computador e executadas pelo computador faz com que o computador reali-

ze o método, como definido pelo menos em uma das reivindicações anteriores.

13. Sistema baseado em computador para gerar dado de saída, em particular, sob a forma de texto legível por humano, que compreende

5 um conjunto de arquivos executáveis,
 uma estrutura repositória e
 uma estrutura de tabela, em que
 a estrutura repositória e a estrutura de tabela são associadas
uma à outra,

10 e
 uma interface gráfica de usuário (GUI, 101) que é adaptada para
 executar os arquivos executáveis,
 ler dados e/ou escrever dados na estrutura de tabela, e
 proporcionar o dado de saída as texto legível por humano,
15 o sistema baseado em computador que compreende adicional-
mente:

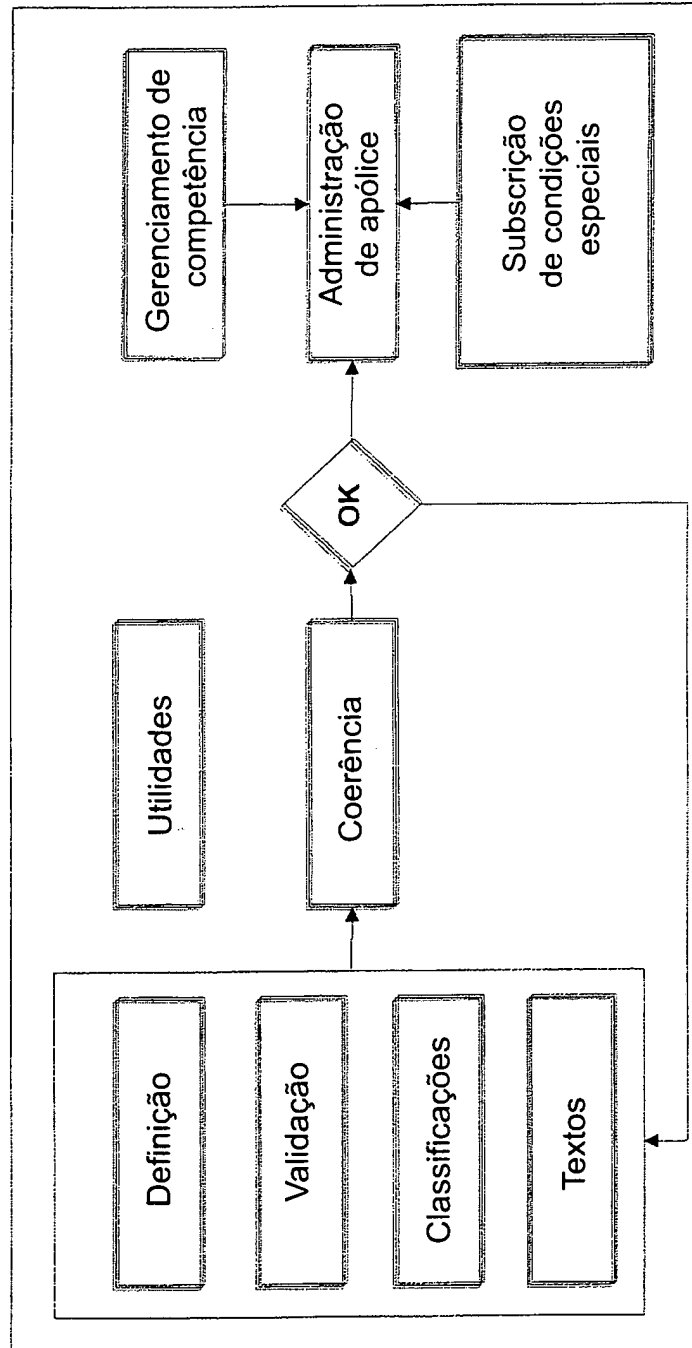
 uma interface de cliente para definir uma alteração dos arquivos
 executáveis em um lado de cliente,
 um dispositivo de servidor para
20 adaptar os arquivos executáveis, de acordo com a definição da
alteração em um lado de servidor,

 introduzir pelo menos uma elemento repositório adicional na es-
trutura repositória no lado de servidor, em que pelo menos um elemento re-
positório adicional corresponde à adaptação dos arquivos executáveis

25 e
 adaptar a estrutura de tabela no lado de servidor, que corres-
ponde ao elemento repositório adicional da estrutura repositória,
e em que

 a dita GUI (101) é adaptada para acessar a estrutura de tabela adaptada no
30 lado de servidor e/ou para executar um ou mais arquivos executáveis no lado
de servidor, enquanto o esboço visual da dita GUI (101) é substancialmente
mantido.

FIG. 1b



Company		System	
User's name Operations Scenario User's group Errors Corporate lab Errors log Structures Administrator 		V R 	
Validation definition - Validation modification			
Validation data Name PLAS		Effective date 01/01/1900	
Description IF TRADE IS EXC, PL, ON NS		Expiration date 31/12/3999	
Logical condition data			
Sequence (		Operator)	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	
Op. T 		Op. F 	
Operand 		Operand 	

FIG. 2

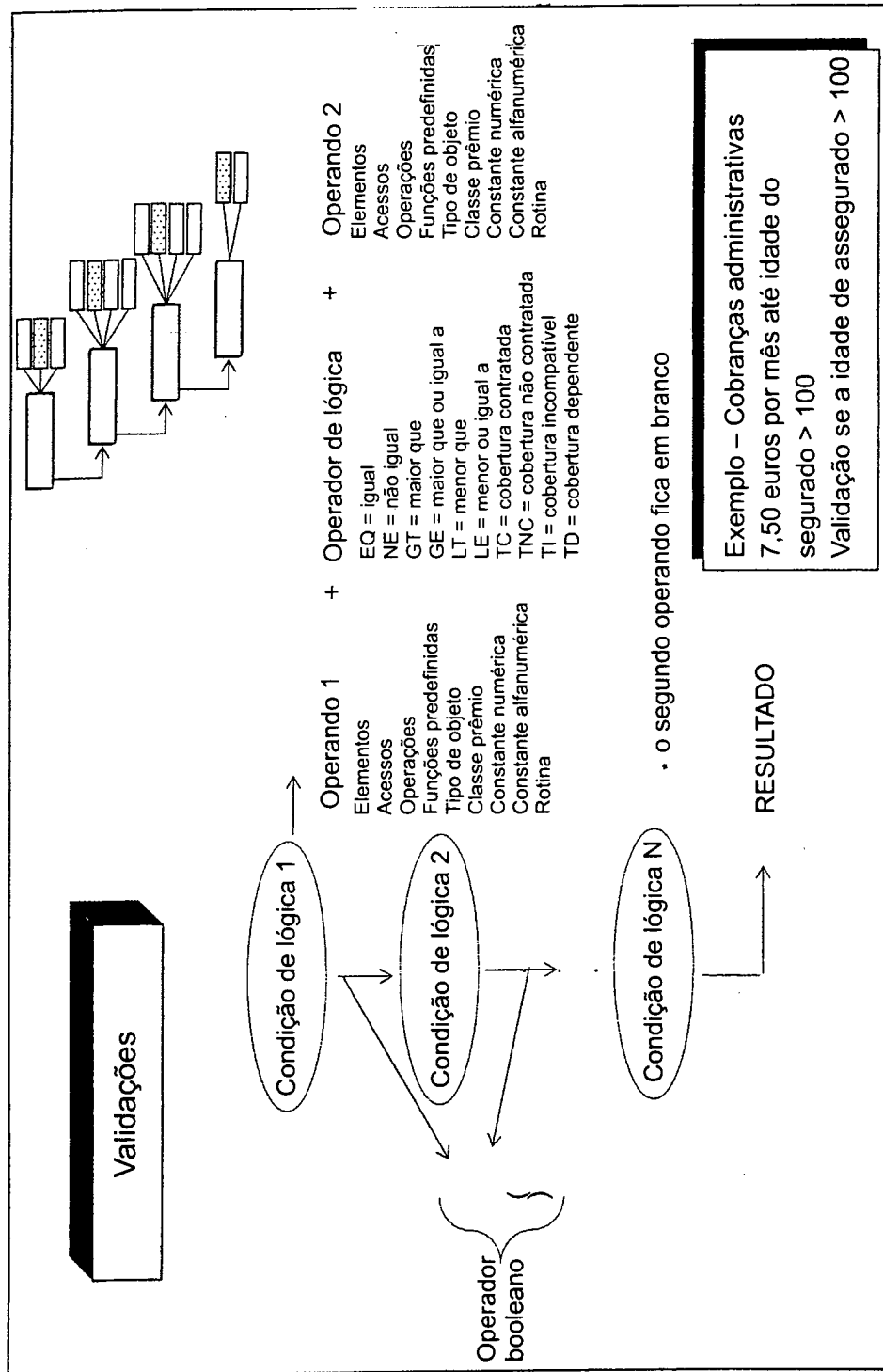


FIG. 3

Company Insurance Solution

Commercial Policy Claims Administration Co-Reinsurance System Window Exit

Prod Cont. Tables Structures Documents Admin Tasks Administrator Tools Help

Validation data

Name: SUM CONTENTS 4 > LIM

Description: SUM CONTENTS 4 > LIMIT

Operator: HIGHER

Operand: SUM CONTENTS 4

Logical condition data:

- Order: ()
- Op1: ()
- Op2: ()
- Op3: ()
- Op4: ()
- Op5: ()
- Op6: ()
- Op7: ()
- Op8: ()
- Op9: ()
- Op10: ()
- Op11: ()
- Op12: ()
- Op13: ()
- Op14: ()
- Op15: ()
- Op16: ()
- Op17: ()
- Op18: ()
- Op19: ()
- Op20: ()
- Op21: ()
- Op22: ()
- Op23: ()
- Op24: ()
- Op25: ()
- Op26: ()
- Op27: ()
- Op28: ()
- Op29: ()
- Op30: ()
- Op31: ()
- Op32: ()
- Op33: ()
- Op34: ()
- Op35: ()
- Op36: ()
- Op37: ()
- Op38: ()
- Op39: ()
- Op40: ()
- Op41: ()
- Op42: ()
- Op43: ()
- Op44: ()
- Op45: ()
- Op46: ()
- Op47: ()
- Op48: ()
- Op49: ()
- Op50: ()
- Op51: ()
- Op52: ()
- Op53: ()
- Op54: ()
- Op55: ()
- Op56: ()
- Op57: ()
- Op58: ()
- Op59: ()
- Op60: ()
- Op61: ()
- Op62: ()
- Op63: ()
- Op64: ()
- Op65: ()
- Op66: ()
- Op67: ()
- Op68: ()
- Op69: ()
- Op70: ()
- Op71: ()
- Op72: ()
- Op73: ()
- Op74: ()
- Op75: ()
- Op76: ()
- Op77: ()
- Op78: ()
- Op79: ()
- Op80: ()
- Op81: ()
- Op82: ()
- Op83: ()
- Op84: ()
- Op85: ()
- Op86: ()
- Op87: ()
- Op88: ()
- Op89: ()
- Op90: ()
- Op91: ()
- Op92: ()
- Op93: ()
- Op94: ()
- Op95: ()
- Op96: ()
- Op97: ()
- Op98: ()
- Op99: ()
- Op100: ()

Script: Compliance/Clauses

Informational: Nothing

Close

user name 21.01.2000 08:30 PM

FIG. 4

Sesión A - [32 x 80]
 Archivo Edición Vista Comunicación Acciones Ventana Ayuda

Process Options Utilities Help

FM/DB2 (DBAI)
 Command ==>

TABLE AISDV1 KTPM34T

1 of 1
 Scroll CSR
 Format SNGL
 in Row 1

Ref	Key	Column	Type(len)	Data
N1	U	CDVAL34T	CH(10)	0000006619
N2	UN	CTSEC34T	DEC(5,0)	1
N3		FEINI34T	DEC(8,0)	19000101
N4		FEFIN34T	DEC(8,0)	99991231
N5	N	NOVALIDA	CH(20)	SUM CONTENTS 4 > LIM
N6	N	DSVALIDA	CH(50)	SUM CONTENTS 4 > LIMIT
N7		CDUSUARI	CH(10)	UAISS19
N8		CDEMPUSU	CH(4)	0001
N9		NOPROGRA	CH(8)	KTPLMM12
N10		TSULIMOD	TIMESTAMP	2007-02-14-13.26.30.982414
***		End of record	***	

F1=Help F2=Zoom F3=Exit F4=CRetrie v F5=RFInd F6=RChange
 F7=Up F8=Down F9=Swap F10=Previous F11=Next F12=Cancel

MA a R 04/015
 Conectado a un sistema principal/servidor remoto 192.168.1.1 utilizando una agrupación de TC40031 y un puerto 23

FIG. 5a

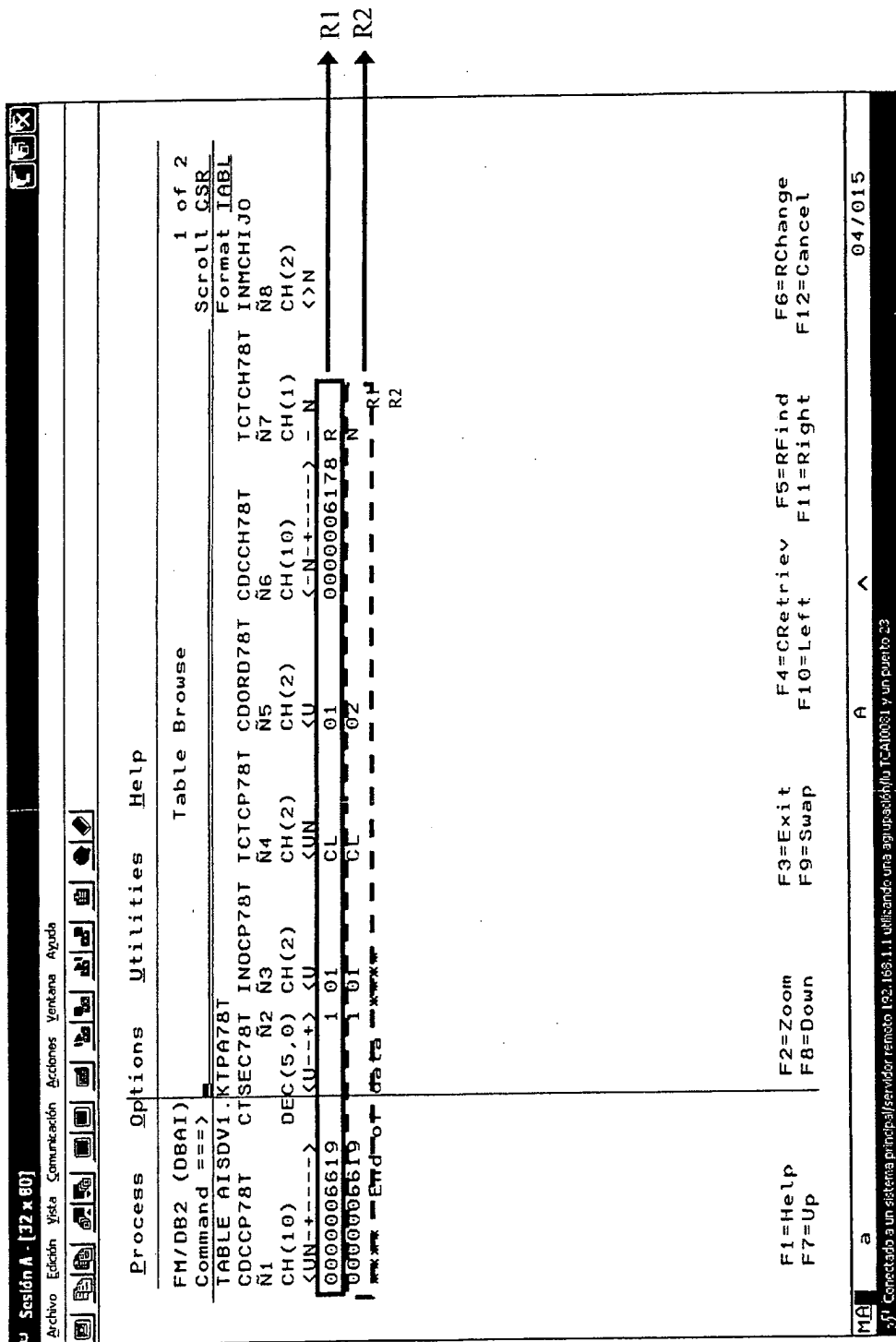




FIG. 6

Sesión A - [32 x 60]

Archivo Edición Vista Comunicación Acciones Ventana Ayuda

File	Edit	Confirm	Menu	Utilities	Compilers	Test	Help
VIEW	MAIS.LIBPRO.COPYS(TP3CCYAU) - 01.00						Columns 00001 00072
Command ==>							Scroll ==> CSR

```

063435      MOVE 'F' TO IR-EX-SENSE
063436          IF USR-0000006178-00 >
063437              2500000
063438          SET USV-0000006619-000-TRUE TO TRUE
063439          MOVE T-TITLE TP-EY-SENSE
063440          END-IF.
```

```

PERFORM TR-ININSERT-KTPA84T
THRU TR-ININSERT-KTPA84T-EXIT.
0000001092000000661900000101-X.
--EXIT.--
0000001092000000667400000101-E.
```

```

SET USV-0000006674-000-FALSE TO TRUE
MOVE WSE-P-CDBROCOU TO
WSP-E-0000005534-001 TO
MOVE WSE-P-CDBROCOU TO
WSP-E-0000005534-001 TO
MOVE WSE-P-CDBROCOU TO
WSP-E-0000005534-001 TO
MOVE WSE-P-CDBROCOU TO
WSP-E-0000005534-001 TO
MOVE WSE-P-CDPRODIE TO
WSP-E-0000005534-002 TO
MOVE WSE-P-CDPRODIE TO
WSP-E-0000005534-002 TO
MOVE WSE-P-CDPRODIE TO
WSP-E-0000005534-002 TO
MOVE WSE-P-CDPRODIE TO
WSP-E-0000005534-002 TO
```

```

F2=Split F3=Exit F5=Rfind F6=Rchange F7=Up
F9=Swap F10=Left F11=Right F12=Cancel
```

04/015

MÁ a

Conectado a un sistema principal/servidor remoto 192.168.1.1 utilizando una agrupación de puertos TCP/IP y un puerto 23

End
Lines

Company Insurance Solution				System Administration			
Commercial	Policy	Claims	Administration	Co-Reinsurance	System	Window	Exit
Prod Conf.	Tables	Structures	Documents	Admin Tasks	Administrator	Tools	Help
--Access definition-- --Access query-- --Access data--				--Access type-- Corporate Table			
ACC_SIC_2ND		Access to SIC table by 2nd Activity Level, as 3rd					
Table code--		Table description--		Non-Date Access--			
KTPTSIT		SIC table		No			
-Key-		-Results-		-Operator-		-Operator name / Constant-	
Beginning date		Beginning date		LOWER OR EQUAL		RATING DATE	
-Key-		-Key-		-Operator type-		-Close-	
Beginning date		Beginning date		LOWER OR EQUAL			
-Format-		-Format-		-Operator-		-Operator-	
DEC, 8, 0		DEC, 8, 0		LOWER OR EQUAL		RATING DATE	
CHAR, 3		CHAR, 3		EQUAL		Broker Country	
CHAR, 8		CHAR, 8		EQUAL		2. ACTIVITY LEVEL	
Activity code 2nd level		Activity code 2nd level		EQUAL		ALPHANUMERIC CONSTA...	
Activity code 3rd level		Activity code 3rd level		EQUAL			
-Script-		-Script-		-Informational-		-Close-	
Compliance/Clauses		Compliance/Clauses		Nothing.			
Nothing.		Nothing.		Nothing.			
-Val. end. date-		-Val. end. date-		-Val. st. date-		-Val. end. date-	
31/12/9999		31/12/9999		01/01/1900		31/12/9999	
-Query by date-		-Query by date-		-Query by date-		-Query by date-	
/ /		/ /		/ /		/ /	

FIG. 7b

Company Insurance Solution

Commercial Policy Claims Administration Co-Reinsurance System Window Exit

Prod Conf. Tables Structures Documents Admin Tasks Administrator Tools Help

Insurance Solution

Prof Vol

Year Policies

-Access definition- **-Access query-**

-Access data-

-Access name- ACC_SIC_2ND

-Access description- Access to SIC table by 2nd Activity Level, as 3rd

-Access Type- Corporate Table

-Table code- KPTST

-Table description- SIC table

-Non-Data Access- No

-Query by date- / /

-Val. st. date- 01/01/1900

-Val. end. date- 31/12/9999

-Keys- **-Results-**

-Key's criteria-

-Key- Beginning date

-Operator- LOWER OR EQUAL

-Operand name / Constant- RATING DATE

-Close-

-Key-	-Operand-	-Operator-	-Operand type-
Activity code 3rd Level	CHAR, 8	EQUAL	ALPHANUMERIC CONSTA...
Corporate table currency code	CHAR, 3	EQUAL	ELEMENT
Commercial product code	CHAR, 10	EQUAL	POLICY CURRENCY
Technical product code	CHAR, 10	EQUAL	COMM. PRODUCT
			TECH. PRODUCT

-Script- Nothing

-Compliance/Clauses- Nothing

-Informational- Nothing

-Close-

user name 21/01/2008 06:23 PM

FIG. 8a

Company Insurance Solution

Commercial Policy Claims Administration
Prod Conf. Tables Structures Documents
Admin. Tasks Administrator Tools Help

Prof. Vol.
Year: Policies:

Insurance Solution

Access definition

Access data

Access query

Access name

ACC_SIC_2ND

Access description

Access to SIC table by 2nd Activity Level, as 3rd

Access Type

Corporate Table

Table code

KIPTSIT

Table description

SIC table

Non-Data Access

No

Query by date

Val. st. date

01/01/1900

Val. end date

31/12/9999

Keys

Results

Code

CDUSPROD

POBULBR

POLIPRRW

POLUSEXP

POLIPRON

POLIPRPA

POLLEMP

POLLEMON

POLIFUEL

Name

US product covered

Building Base Rate used in Base P

Rate % Products liability rest of Ind

Rate % Products US exports

Rate % Products liability DINC

Rate % 1st party recall liability

Rate % third party

Rate % Public liability DINC limit

Rate % public liability

Code

CDACTGRU

COMBCLAS

CDRETORN

CDTHCLAS

CDUSPROD

CDVYCLAS

IMBASLIM

IMPROLIM

POBULBR

Name

Group the 2nd level activity belong

MB Class

Return code

Theft Class

US product covered

VV Class

Base Capacity limit

Protected Capacity Limit

Building Base Rate used in Base P

Script

Nothing

Compliance/Clauses

Nothing

Informational

Nothing

Close

user name

21/01/2008

06:23 PM

Company Insurance Solution		System Administration	
Commercial Policy Claims Administration	Co-Reinsurance System Window Exit		
Prod Conf. Tables Structures Documents	Admin Tasks Administrator Tools Help		

-Access definition- -Access query-

-Access data-

-Access name-

ACC_SIC_2ND

-Table code-

TPTSTT

-Access description-

Access to SIC table by 2nd Activity Level, as 3rd

-Table description-

SIC table

-Non-Data Access- No

-Access Type-

Corporate Table

-Query by date-

Val. st. date: 01/01/90 Val. end. date: 31/12/9999

-Keys-	-Results-
-Code- POLEMPD POBUSINT POSTOCBR POPRATE PODOACMD POPAACMD POGACAMD POTRACMD CDBGLAS	-Name- Rate % acc pollution BI Base Rate Stock base rate Rate % PI Activity Modifier D&O Activity Modifier PA Activity Modifier Crime Activity Modifier Travel MB Class

-Code-	-Name-
POLUSINT	BI Base Rate
POGRACMD	Activity Modifier Crime
PODOACMD	Activity Modifier D&O
POLEMON	Rate % Public liability DINC limit
POLEMPO	Rate % acc pollution
POLIEMTP	Rate % third party
POLIPRON	Rate % Products liability DINC
POLIPRPA	Rate % 1st party recall liability
POLIPRRW	Rate % Products liability rest of the

-Script-

-Compliance/Clauses-

Nothing.

-Informational-

Nothing.

User name
21/01/2008
08:24 PM

Process		Options	Utilities	Help
FM/DB2 (DBAI)			Table Browse	1 of 1 Scroll CSR Format SNGL
Command ==>				
TABLE AISDV1.KTPH38T		Top Line is 1 of 16 in Row 1		
Ref Key	Column	Type(len)	Data	
N1 U	CDACCESO	CH(10)	0000001280	
N2 UN	CTSEQ38T	DEC(5,0)	1	
N3	FEINI38T	DEC(8,0)	19000101	
N4	FEFIN38T	DEC(8,0)	99991231	
N5 N	NOACCESO	CH(20)	ACC_SIC_2ND	
N6 N	DSACCESO	CH(50)	Access to SIC table by 2nd Activity Level , as 3rd	
N7	CDTYTAB	CH(2)	CT	
N8	TCFUNACC	CH(2)	SE	
N9	TCOPEACC	CH(1)	S	
N10	NOTABLAC	CH(10)	KIPTISIT	
N11	CDSQL	CH(1)	N	
N12	CDUSUARI	CH(10)	UAISS28	
N13	CDEMPUSU	CH(4)	0001	
N14	NOPROGRA	CH(8)	KTPLMM16	
N15	TSLTMOD	TIMESTAMP	2007-01-29-20.46.46.428515	
***	End of record	***		
F1=Help F7=Up	F2=Zoom F8=Down	F3=Exit F9=Swap	F4=CRetrieval F10=Previous	F5=RFind F11=Next F6=RChange F12=Cancel

FIG. 10a

Sesión A - [32 x 80]

Archivo

Edición

Vista

Comunicación

Acciones

Ventana

Ayuda

Process

Options

Utilities

Help

FM/DB2 (DBAI)

Command ==>

Table Browse

1 of 24

Scroll CSR

Format Table

TABLE AISDWH_VIPD30T		TABLE AISDWH_VIPD30T	
CDACCESO	CTSEQ38T	CDRES30T	NORA30T
N1	DEC(5,0)	CH(8)	N4
CH(10)	CH(8)	CHARACTER(50)	
00000001280	<U-->	<U-->	Group the 2nd level activity belongs to
00000001280	1	CDMBCLAS	MB Class
00000001280	1	CDRETORN	Return code
00000001280	1	CDTHCLAS	Theft Class
00000001280	1	CDUSPROD	US product covered
00000001280	1	CDVVCLAS	VV Class
00000001280	1	IMBASLIM	Base Capacity limit
00000001280	1	IMPROLIM	Protected Capacity limit
00000001280	1	POBUILAR	Building Base Rate used in Base Perils
00000001280	1	POBUSINT	BI Base Rate
00000001280	1	POCRACMD	Activity Modifier Crime
00000001280	1	PODOACMD	Activity Modifier D&O
00000001280	1	POLIEMDN	Rate % Public liability DINC limit
00000001280	1	POLIEMPO	Rate % acc pollution
00000001280	1	POLIEMTP	Rate % third party
00000001280	1	POLIPRDN	Rate % Products liability DINC
00000001280	1	POLIPRPA	Rate % 1st party recall liability
00000001280	1	POLIPRRW	Rate % Products liability rest of the world
00000001280	1	POLIPUBL	Rate % public liability
00000001280	1	POLUSEXP	Rate % Products US exports
00000001280	1	POPACMD	Activity Modifier PA
F1=Help	F2=Zoom	F3=Exit	F4=CRetriev
F7=Up	F8=Down	F9=Swap	F10=Left
		F5=RFind	F11=Right
		F6=RChange	F12=Cancel

R1

R2

etc

MA

a

04/015

Conectado a un sistema principal/secondary remoto 192.168.1.1 utilizando una agrupación de TCAD081 y un puerto 23

FIG. 10b

16/59

Sesión A - [32 x 80]		22 of 24	
Archivo Edición Vista Comunicación Acciones Ventana Ayuda		Scroll CSR	
Process Options Utilities Help		Format TABL	
FM/DB2 (DBAI)	Table Browse		
Command ==>			
TABLE AISDV1.KTPA30I			
CDACCESO	CISEQ38I	CDRES30I	NORA30I
N1	N2 N3	N4	
CH(10)	DEC(5,0)	CH(8)	CHARACTER(50)
<U-->	<U-->	<U-->	<U-->
0000001280	1 POPIRATE	Rate % PI	
0000001280	1 POSTOCBR	Stock base rate	
0000001280	1 POTRACMD	Activity Modifier	Travel
****	End of data	****	
F1=Help	F2=Zoom	F3=Exit	F4=CRetrieval
F7=Up	F8=Down	F9=Swap	F10=Left
		F5=RFind	F6=RChange
		F11=Right	F12=Cancel
a		04/015	
Conectado a un sistema principal/servidor remoto 192.168.1.1 utilizando una agrupación TCPIP0081 y un puerto 23			

FIG. 11a

Sesión A - [32 x 80]

Archivo Edición Vista Comunicación Acciones Ventana Ayuda

Process Options Utilities Help
0 of 7

FM/DB2 (DBAI)
Scroll CSR

Command ==>
Format IABL

TABLE AISDV1.KTPA37T		NOCLAVEA		ICOPERCL		CDOPERAN		VALOR37T	
CDACCESO CTSEQ38T		N4		N5		N6		N7	
N1		CH(8)		CH(8)		CH(10)		CHARACTER(20)	
CH(10)		DEC(5,0)		CH(2)		CH(10)		CHARACTER(20)	
<U--+->		<U--+->		<U--+->		<N--+->		<--+->	
**** Top of data ****		FEINICIO <=		FETARIFA		CDBROCOU		TCPACTI2	
0000001280		1 01		CDBROCOU =		TCCDMONE		CDPRODIE	
0000001280		1 02		TCACTIV2 =		CDPRODIE			
0000001280		1 03		TCACTIV3 =					
0000001280		1 04		TCCDMONE =					
0000001280		1 05		CDPRODIE =					
0000001280		1 06							
0000001280		1 07							
**** End of data ****									

F1=Help

F7=Up

F2=Zoom

F8=Down

F3=Exit

F9=Swap

F4=CRetrie

F10=Left

F5=RFInd

F11=Right

F6=RChange

F12=Cancel

MA a
04/015

17/59

R1

R2

etc

FIG. 11b

Sesión A - [32 x 80]

Archivo

Edición

Vista

Comunicación

Acciones

Ventana

Ayuda

Process

Options

Utilities

Help

FM/DB2 (DBAI)

Command ==>

Table Browse

0 of 7

Scroll CSR

Format Table

TABLE AISOW1 KIDQ271

TCOPERAN CDRDOANT CDUSUARI

N8 N9 N10 N11 N12 N13

CH(2) CH(8) CH(10) CH(4) CH(8) CH(8) CH(8) CH(8) CH(8) CH(8) CH(8) CH(8)

<N> <---+---> <---+---> <---+---> <---+---> <---+---> <---+---> <---+---> <---+---> <---+---> <---+---> <---+--->

**** Top of data ****

E UAIS28 0001

E UAIS28 0001

E UAIS28 0001

F UAIS28 0001

E UAIS28 0001

E UAIS28 0001

E UAIS28 0001

**** End of data ****

KTPLMM16 2007-01-29-20.46.46.447206

KTPLMM16 2007-01-29-20.46.46.447705

KTPLMM16 2007-01-29-20.46.46.448072

KTPLMM16 2007-01-29-20.46.46.448448

KTPLMM16 2007-01-29-20.46.46.448767

KTPLMM16 2007-01-29-20.46.46.449138

KTPLMM16 2007-01-29-20.46.46.449509

F1=Help

F2=Zoom

F3=Exit

F4=CRetrie v

F5=RFInd

F6=RChange

F7=Up

F8=Down

F9=Swap

F10=Left

F11=Right

F12=Cancel

Má a

04/015

Conectado a un sistema principal/servidor remoto 192.168.1.1 utilizando una agrupación TCAD003 y un puerto 23

18/59

FIG. 12a

Sesión A - [32 x 80]

Archivo

Edición

Vista

Comunicación

Acciones

Ventana

Ayuda

File

Edit

Confirm

Menu

Utilities

Compilers

Test

Help

VIEW

MAIS.LIBEPRO.COPYS(IP3CCYAU) - 01.00

Columns 00001 00072

Command ==>

Scroll ==> CSR

061899

000000001280-000001.

IF WS-00000001280-CLA-PRV NOT = WS-00000001280-CLA

MOVE WS-00000001280-CLA TO WS-00000001280-CLA-PRV

EXEC SQL

DECLARE CURSOR EQR

SELECT

CDACTGRU ,

CDMBCLAS ,

CDTHCLAS ,

CDUSPROD ,

CDVVCLAS ,

FEINICIO ,

IMBASLIM ,

IMPROLIM ,

POBUILBR ,

POBUSINT ,

POCRACMD ,

PODOACMD ,

POLIEMDN ,

POLIEMPO ,

POLIPRDN ,

POLIPRPA ,

POLIPRRW ,

POLIPUBL ,

POLUSEXP ,

POPAACMD ,

F3=Exit

F5=Rfind

F6=Rchange

F7=Up

F2=Split

F9=Swap

F10=Left

F11=Right

F12=Cancel

061900

061901

061902

061903

061904

061905

061906

061907

061908

061909

061910

061911

061912

061913

061914

061915

061916

061917

061918

061919

061920

061921

061922

061923

061924

F1=Help

F8=Down

L1

L2

etc

MA

a

05/016

3) Conectado a un sistema principal/servidor remoto 192.168.1.1 utilizando una agrupación de TCA0001 y un puerto 23

FIG. 12b

Sesión A - [32 x 80]

Archivo Edición Vista Comunicación Acciones Ventana Ayuda

File Edit Confirm Menu Utilities Compilers Test Help

VIEW MAIS.LIBPRO.COPYS(IP3CCYAU) - 01.00 Columns 00001 00072

Command ==> CSR

```

061925
061926
061927
061928
061929
061930
061931
061932
061933
061934
061935
061936
061937
061938
061939
061940
061941
061942
061943
061944
061945
061946
061947
061948
061949
061950
F1=Help
F8=Down
F2=Split
F9=Swap
F10=Left
F11=Right
F12=Cancel
F6=Rchange
F7=Up

FROM KTPTSIT
WHERE FEINICIO <= :WSA-E-00000001280FETARIFA AND
CDBROCOU = :WSA-E-00000001280CDBROCOU AND
TCACTIV2 = :WSA-E-00000001280TCACTI2 AND
TCACTIV3 = :WSA-F-00000001280TCACTIV3 AND
TCCDMONE = :WSA-E-00000001280TCCDMONE AND
CDPRODCO = :WSA-E-00000001280CDPRODCO AND
CDPROOTE = :WSA-E-00000001280CDPROOTE
ORDER BY FEINICIO DESC
END-EXEC
EXEC SQL OPEN CC0000000128000001 END-EXEC
IF SQLCODE < 0
MOVE WC-SQL-ERROR TO CDERR
MOVE WC-OPEN-CURSOR-1 TO DSARGU1
MOVE 'KTPTSIT01280' TO DSARGU2
MOVE SQLCODE TO DSADERR
PERFORM 9999-ERROR-MANAG THRU 9999-ERROR-MANAG-EXIT
END-IF
EXEC SQL FETCH CC0000000128000001 INTO
:WSA-00000001280-CDACTGRU,
:WSA-00000001280-CDMBCLAS,
:WSA-00000001280-CDTHCLAS,
:WSA-00000001280-CDUSPROD,
F3=Exit F5=Rfind F6=Rchange F7=Up
F2=Split F10=Left F11=Right F12=Cancel
F9=Swap

```

04/015

POPIRATE ,

POSTOCBR ,

POIRACMD

L1

L2

etc

Fixed sql lines

MA

a

Corrección a un sistema principal/servidor remoto 192.168.1.1 utilizando una agrupación TCAL00001 y un puerto 23

FIG. 12c

Sesión A - [32 x 80]

Archivo

Edición

Vista

Comunicación

Acciones

Ventana

Ayuda

File

Edit

Confirm

Menu

Utilities

Compilers

Test

Help

VIEW

Command ===>

MAIS.LIBPRO.COPYS(IP3CCYAU) - 01.00

Columns 00009 00080

Scroll ==> CSR

```

061951      :WSA-00000001280-CDVVCLAS,
061952      :WSA-00000001280-FEINICIO,
061953      :WSA-00000001280-IMBASLIN,
061954      :WSA-00000001280-IMPROLIN,
061955      :WSA-00000001280-POBUILBR,
061956      :WSA-00000001280-POBUSINT,
061957      :WSA-00000001280-POCRACMD,
061958      :WSA-00000001280-PODOACMD,
061959      :WSA-00000001280-POLIEMDN,
061960      :WSA-00000001280-POLIEMPO,
061961      :WSA-00000001280-POLIEMTP,
061962      :WSA-00000001280-POLIPRDN,
061963      :WSA-00000001280-POLIPRPA,
061964      :WSA-00000001280-POLIPRRW,
061965      :WSA-00000001280-POLIPUBL,
061966      :WSA-00000001280-POLUSEXP,
061967      :WSA-00000001280-POPAACMD,
061968      :WSA-00000001280-POPIRATE,
061969      :WSA-00000001280-POSTOCBR,
061970      :WSA-00000001280-POTRACMD
END-EXEC
EVALUATE SOURCE
WHEN 0 MOVE 0 TO WSA-00000001280-CDRETORN
WHEN +100
MOVE WC-SQL-100 TO COERR
MOVE '00000001280' TO OSARGU1
F2=Split      F3=Exit      F5=Rfind      F6=Rchange      F7=Up
F9=Swap      F10=Left      F11=Right      F12=Cancel
F8=Down

```

MA

a

04/015

Conectado a un sistema principal/servidor remoto 192.168.1.1 utilizando una agrupación de TCAD0001 y un puerto 23

22/59



FIG. 13

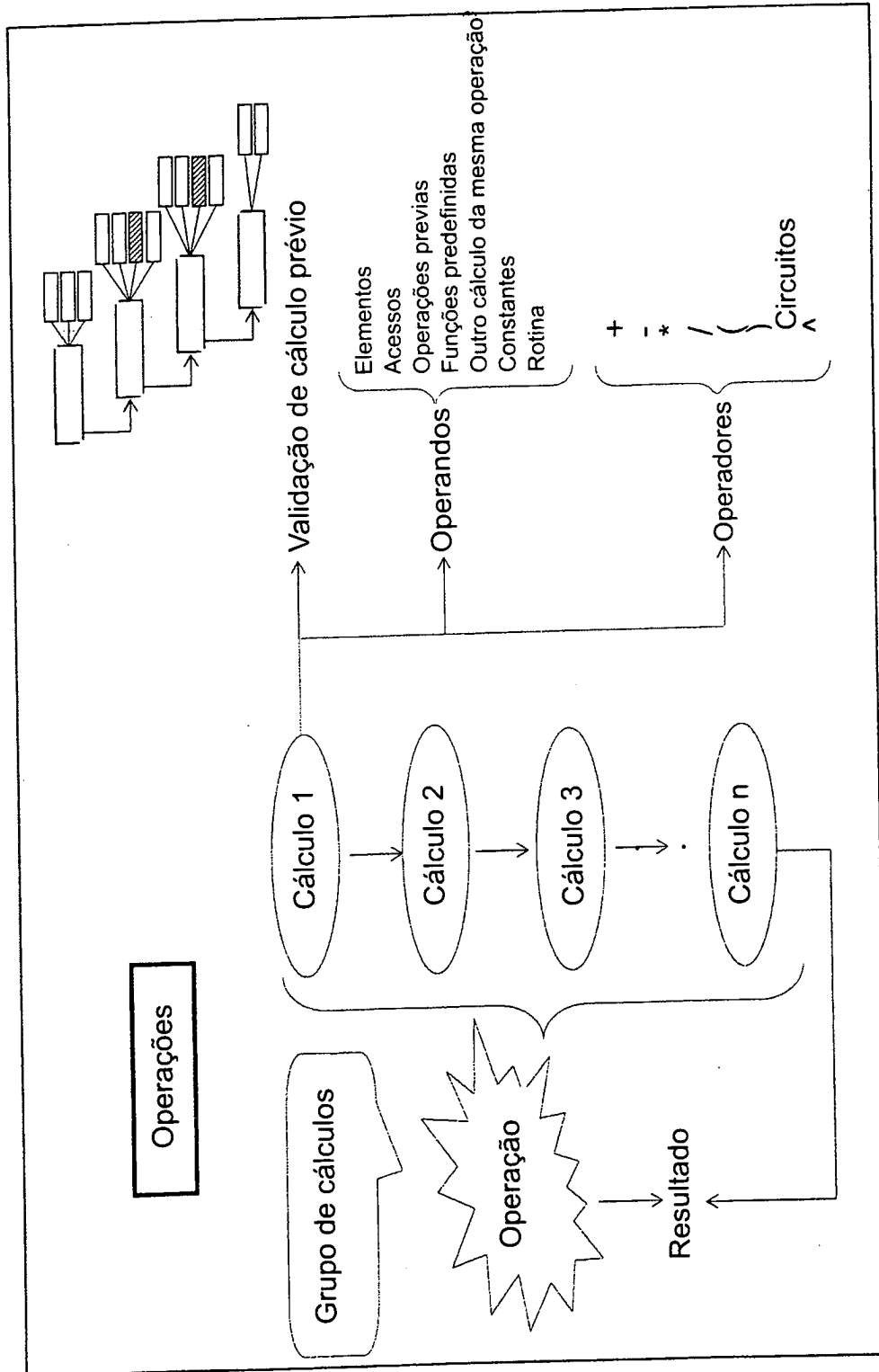


FIG. 14a

Company Insurance Solution

Commercial Policy Claims Administration Co-Reinsurance System Window Exit

Prod Conf. Tables Structures Documents Admin Tasks Administrator Tools Help

System Administration

Prof Vol

Year: Policies: Val

Calculations - Operation query

-Name- DP_DEFAULT_B -Description- DP_DEFAULT_B -Val. st. date- 01/01/1900 -Val. end. date- 31/12/9999 -Query d.- / /

-Data type- INTERMEDIATE VALUE 00: -Operation type- USER -Insurance ty... -Pred. func. type- -Amount- -Round- ☐ -Truncate- ☒

Calculation data

-Order- 01 -Logical condition- SUM CONTENTS 4 > LIM -Situation- FALSE -Calculation name- C1 -Format- DEC 15, 3

-Calculation- (SUM CONTENTS 4) * (0.1)

-Formals... -Loop cond - -Close-

Order

Order	Condition	Situation	Calculation name
01	SUM CONTENTS 4 > LIM	FALSE	C1
05	SUM CONTENTS 4 > LIM	TRUE	C2
10	SUM CONTENTS 4 > LIM	C3	C3

Script

-Compliance/Clauses- Nothing

-Information- Nothing

-Close-

user name 21/01/2008 06:05 PM

FIG. 14b

Company Insurance Solution

Commercial Policy Claims Administration System Window Exit

Prod.Conf. Tables Structures Documents Admin.Tasks Administrator Tools Help

Insurance Solution

Year: Policies: Val

Calculations - - Operation query -

-Name- OP_DEFAULT_8 -Description- OP_DEFAULT_8 -Val. st. date- 01/01/1900 -Val. end. date- 01/01/2999 -Query d- 1/1

-Data type- INTERMEDIATE VALUE 00: -Operation type- USER -Insurance ty- -Pred func. type- -Amount- -Format- DEC, 15, 3

-Calculation data-

-Order- 05 -Logical condition- SUM CONTENTS 4 > LIM -Situation- TRUE -Calculation name- C2 -Round- ☐ Round ☒ Truncate

-Calculation- 2500000

-Operator-

-Formets... -Loop cond... -Close-

-Order-	-Condition-	-Situation-	-Calculation name-
01	SUM CONTENTS 4 > LIM	FALSE	C1
05	SUM CONTENTS 4 > LIM	TRUE	C2
10			C3

-Script- -Compliance/Clauses- -Informational- Nothing

user name 21/01/2008 06:05 PM

FIG. 14C

Company Insurance Solution

Commercial Policy Claims Administration Lo-Reinsurance System Window Exit

Prod Conf. Tables Structures Documents Admin Tasks Administrator Tools Help

Insurance Solution

Year: Policies: Val: end: ?

-Calculations- -Operation query- -Val. st. date- 31/01/1900 -Val. end. date- 31/12/9999 -Query d.- / /

-Name- OP_DEFAULT_B -Description- OP_DEFAULT_B -Insurance ty.- -Pred. func. type-

-Data type- INTERMEDIATE VALUE 001 -Amount- -Situation- TRUE -Calculation name- C3 -Format- DEC, 15, 3 -Round- ☐ -Round- ☒ -Truncate-

-Calculation data-

-Order- 10 -Logical condition- ☒ -Calculation- [C1] + [C2] -Opnd.- ☒ -Operator- + -Close-

-Formets.- -Loop cond.- -Close-

-Order-	-Condition-	-Situation-	-Calculation name-
01	SUM CONTENTS 4 > LIM	FALSE	C1
05	SUM CONTENTS 4 > LIM	TRUE	C2
10			C3

-Script- -Compliance/Clauses- -Informational- Nothing

user name 21/01/2008 05:05 PM -Close-

FIG. 14d

Company - System

Users'mainte... Users'group ... Functions'mal... Corporate tab... Structures ...
 Operations'm... Scenario/Ope... Errors' mainte... Errors'log det... Administrator ...

Operations definition - New Operation

Name: IPT Description: INSURANCE PREMIUM TAX Effective date: 01/01/1900 Expiration date: 31/12/9999
 Data type: IMPURSTO IPS ... Operation type: STANDARD Insurance type: Amount Pred.Funct.Type:
☒ Operation ☐ Predefined func.

Calculation data:
 Logical condition: Situation: Calculation name: Format: Round up:
 01: [A: IPT: PREMIUM RATE] / (100) DEC, 8, 0 Truncate
 Calculation: (10: PREMIUM BEFORE TAX) * (A: IPT: PREMIUM RATE) / (100)
 Operd. Operator

Formals? Loop Cond. Close

Sequence	Condition	Situation	Calculation name
01			INSURANCE PREM TAX

Refresh Close

03 Z7777777 22H02002 18.54

Process		Options		Utilities		Help	
FM/DB2 (DBA1)				Table Browse		1 of 1	
Command ==>						Scroll CSR Format SNGL	
TABLE AISDV1.KTPM56T				Top Line is 1 of 16 in Row 1			
Ref Key	Column	Type(len)	Data				
N1 U	CDOPE56T	CH(10)	00000006092				
N2 UN	CTSEC56T	DEC(5,0)	1				
N3	FEINJ56T	DEC(8,0)	19000101				
N4	FEFIN56T	DEC(8,0)	99991231				
N5 N	NOOPERAC	CH(20)	OP_DEFAULT_8				
N6 N	DSOPERAC	CH(50)	OP_DEFAULT_8				
N7	INIMPOR	CH(1)	N				
N8	TCOPERAC	CH(1)	U				
N9	TCIPSTD	CH(1)					
N10	TCIPIFUN	CH(1)					
N11	CDELEM	CH(8)	VAINT007				
N12	CDUSUARI	CH(10)	UAISS19				
N13	CDEMPUSU	CH(4)	0001				
N14	NOPROGRA	CH(8)	KTPLMM21				
N15	TSULTMOD	TIMESTAMP	2007-02-16-13.02.22.027116				
***	End of record	***					
F1=Help F7=Up	F2=Zoom F8=Down	F3=Exit F9=Swap	F4=CRetriev F10=Previous	F5=RFind F11=Next	F6=RChange F12=Cancel		

FIG. 16a

29/59

Sesión A - [32 x 80]

Archivo

Edición

Vista

Comunicación

Acciones

Ventana

Ayuda

Process

Options

Utilities

Help

FM/DB2 (DBAI)

Command ==>

TABLE AISDV1 KTPA57T

CDOPES6T CTSEC56T INCAL57T NOCALCUL

N1 N2 N3 N4

CH(10) DEC(5,0) CH(2) CHARACTER(20)

..<UN>.....<UN>.....<UN>.....<UN>.....

00000006092 1 01 C1 0000006619 F I

00000006092 1 05 C2 0000006619 V I

00000006092 1 10 C3 0000000000 I

***** EPR 0 *****

1 of 3

Scroll CSR

Format TABL

INSV57T INREDEC

N6 N7

CH(1) CH(1)

R1

R2

R3

F1=Help

F2=Zoom

F3=Exit

F4=CRetrieval

F5=RFind

F6=RChange

F7=Up

F8=Down

F9=Swap

F10=Left

F11=Right

F12=Cancel

MA

a

04/015

Conectado a un sistema principal/servidor remoto 192.168.1.1 utilizando una agrupación de puertos TCAI0001 y un puerto 23

Sesión A - [32 x 80]

Archivo Edición Vista Comunicación Archivos Ventanas Ayuda

Process	Options	Utilities	Help
FM/DB2 (DBAI) Command ===>		Table Browse	1 of 3 Scroll CSR Format TABL
<pre> TABLE AISDV1.KIPA571 TCFOR571 COUSUARI N8 N9 CH(4) CH(10) CH(4) CH(8) N10 N12 TIMESTAMP D153 UAIS19 0001 KIPLMM22 2007-02-14-13.35.17.480978 D153 UAIS19 0001 KIPLMM22 2007-02-14-13.36.00.380283 D153 UAIS19 0001 KIPLMM22 2007-02-14-13.36.28.176821 -----END OF DATA----- </pre>			
F1=Help F7=Up	F2=Zoom F8=Down	F3=Exit F9=Swap F4=CRetrieV F10=Left F5=RFInd F11=Right	F6=RChange F12=Cancel

Má a

Conectado a un sistema principal/servidor remoto 192.168.1.1 utilizando una agrupación/lu TC400061 y un puerto 23

04/015

FIG. 17a

Sesión A - [32 x 80]

Archivo

Edición

Vista

Comunicación

Acciones

Ventana

Ayuda

Process

Options

Utilities

Help

Table Browse

0 of 5

Scroll CSR

Format IABL

FM/DB2 (DBAI)

Command ==>

TABLE AISDV1.KIPA78T

CUCCP78T

CTSEC78T

INOC78T

TCICP78T

CDORU78T

CDCC78T

TCIC78T

INMCH78T

N1

CH(10)

<UN-+----->

**** Top of data ****

0000006092

0000006092

0000006092

0000006092

0000006092

0000006092

**** End of data ****

N2 N3

DEC(5,0)

CH(2)

<U--+>

1 01

1 01

1 05

1 10

1 10

N4

CH(2)

<UN

CC

CC

CC

CC

CC

N5

CH(2)

<U

01

02

01

02

N6

CH(10)

<-N-+----->

0000006178

01

05

N7

CH(1)

- N

R

N

N

C

N8

CH(2)

<>N

F1=Help

F7=Up

F2=Zoom

F8=Down

F3=Exit

F9=Swap

F4=CRetriev

F10=Left

F5=RFind

F11=Right

F6=RChange

F12=Cancel

MA

a

04/015

3) Conectado a un sistema principal/servidor remoto 192.168.1.1 utilizando una agrupación TC400031 y un puerto 23

FIG. 18a

Sesión A - [32 x 80]

Archivo

Edición

Vista

Comunicación

Acciones

Ventana

Ayuda

File

Edit

Confirm

Menu

Utilities

Compilers

Test

Help

VIEW

MAIS.LIBEPRO.COMPY(S(TP3CCYAU) - 01.00

Columns 00001 00072

Command ==>

Scroll ==> CSR

00000001092-

PERFORM 0000000109200000006619-E THRU 0000000109200000006619-X

IF WSV-00000006619-000

PERFORM 000000010920000000609200000101

THRU 000000010920000000609200000101-X

END-IF

PERFORM 0000000109200000006619-E THRU 0000000109200000006619-X

IF WSV-00000006619-000

PERFORM 000000010920000000609200000105

THRU 000000010920000000609200000105-X

END-IF

PERFORM 000000010920000000609200000110

THRU 000000010920000000609200000110-X

INITIALIZE TR-EX-VALUES

REPLACING ALPHANUMERIC DATA BY SPACES

NUMERIC DATA BY ZERO

MOVE 'OPE' TO TR-EX-TYPE OF TR-EX-VALUES

MOVE 'N' TO TR-EX-ROUNDING OF TR-EX-LEVEL

MOVE WSC-00000001092-00000006092-10

TO WSC-00000001092-00000006092

MOVE WSC-00000001092-00000006092

TO TR-EX-RESULT OF TR-EX-OPERATION

PERFORM TR-INSERT-KTPA84T

THRU TR-INSERT-KTPA84T-EXIT.

00000001092-00000006092-000001-X

F3=Exit F5=Rfind F7=Up

F2=Split F10=Left F11=Right F12=Cancel

F9=Swap F8=Down

00000001092-

PERFORM 0000000109200000006619-E THRU 0000000109200000006619-X

IF WSV-00000006619-000

PERFORM 000000010920000000609200000101

THRU 000000010920000000609200000101-X

END-IF

PERFORM 0000000109200000006619-E THRU 0000000109200000006619-X

IF WSV-00000006619-000

PERFORM 000000010920000000609200000105

THRU 000000010920000000609200000105-X

END-IF

PERFORM 000000010920000000609200000110

THRU 000000010920000000609200000110-X

INITIALIZE TR-EX-VALUES

REPLACING ALPHANUMERIC DATA BY SPACES

NUMERIC DATA BY ZERO

MOVE 'OPE' TO TR-EX-TYPE OF TR-EX-VALUES

MOVE 'N' TO TR-EX-ROUNDING OF TR-EX-LEVEL

MOVE WSC-00000001092-00000006092-10

TO WSC-00000001092-00000006092

MOVE WSC-00000001092-00000006092

TO TR-EX-RESULT OF TR-EX-OPERATION

PERFORM TR-INSERT-KTPA84T

THRU TR-INSERT-KTPA84T-EXIT.

00000001092-00000006092-000001-X

F3=Exit F5=Rfind F7=Up

F2=Split F10=Left F11=Right F12=Cancel

F9=Swap F8=Down

00000001092-

PERFORM 0000000109200000006619-E THRU 0000000109200000006619-X

IF WSV-00000006619-000

PERFORM 000000010920000000609200000101

THRU 000000010920000000609200000101-X

END-IF

PERFORM 0000000109200000006619-E THRU 0000000109200000006619-X

IF WSV-00000006619-000

PERFORM 000000010920000000609200000105

THRU 000000010920000000609200000105-X

END-IF

PERFORM 000000010920000000609200000110

THRU 000000010920000000609200000110-X

INITIALIZE TR-EX-VALUES

REPLACING ALPHANUMERIC DATA BY SPACES

NUMERIC DATA BY ZERO

MOVE 'OPE' TO TR-EX-TYPE OF TR-EX-VALUES

MOVE 'N' TO TR-EX-ROUNDING OF TR-EX-LEVEL

MOVE WSC-00000001092-00000006092-10

TO WSC-00000001092-00000006092

MOVE WSC-00000001092-00000006092

TO TR-EX-RESULT OF TR-EX-OPERATION

PERFORM TR-INSERT-KTPA84T

THRU TR-INSERT-KTPA84T-EXIT.

00000001092-00000006092-000001-X

F3=Exit F5=Rfind F7=Up

F2=Split F10=Left F11=Right F12=Cancel

F9=Swap F8=Down

00000001092-

PERFORM 0000000109200000006619-E THRU 0000000109200000006619-X

IF WSV-00000006619-000

PERFORM 000000010920000000609200000101

THRU 000000010920000000609200000101-X

END-IF

PERFORM 0000000109200000006619-E THRU 0000000109200000006619-X

IF WSV-00000006619-000

PERFORM 000000010920000000609200000105

THRU 000000010920000000609200000105-X

END-IF

PERFORM 000000010920000000609200000110

THRU 000000010920000000609200000110-X

INITIALIZE TR-EX-VALUES

REPLACING ALPHANUMERIC DATA BY SPACES

NUMERIC DATA BY ZERO

MOVE 'OPE' TO TR-EX-TYPE OF TR-EX-VALUES

MOVE 'N' TO TR-EX-ROUNDING OF TR-EX-LEVEL

MOVE WSC-00000001092-00000006092-10

TO WSC-00000001092-00000006092

MOVE WSC-00000001092-00000006092

TO TR-EX-RESULT OF TR-EX-OPERATION

PERFORM TR-INSERT-KTPA84T

THRU TR-INSERT-KTPA84T-EXIT.

00000001092-00000006092-000001-X

F3=Exit F5=Rfind F7=Up

F2=Split F10=Left F11=Right F12=Cancel

F9=Swap F8=Down

00000001092-

PERFORM 0000000109200000006619-E THRU 0000000109200000006619-X

IF WSV-00000006619-000

PERFORM 000000010920000000609200000101

THRU 000000010920000000609200000101-X

END-IF

PERFORM 0000000109200000006619-E THRU 0000000109200000006619-X

IF WSV-00000006619-000

PERFORM 000000010920000000609200000105

THRU 000000010920000000609200000105-X

END-IF

PERFORM 000000010920000000609200000110

THRU 000000010920000000609200000110-X

INITIALIZE TR-EX-VALUES

REPLACING ALPHANUMERIC DATA BY SPACES

NUMERIC DATA BY ZERO

MOVE 'OPE' TO TR-EX-TYPE OF TR-EX-VALUES

MOVE 'N' TO TR-EX-ROUNDING OF TR-EX-LEVEL

MOVE WSC-00000001092-00000006092-10

TO WSC-00000001092-00000006092

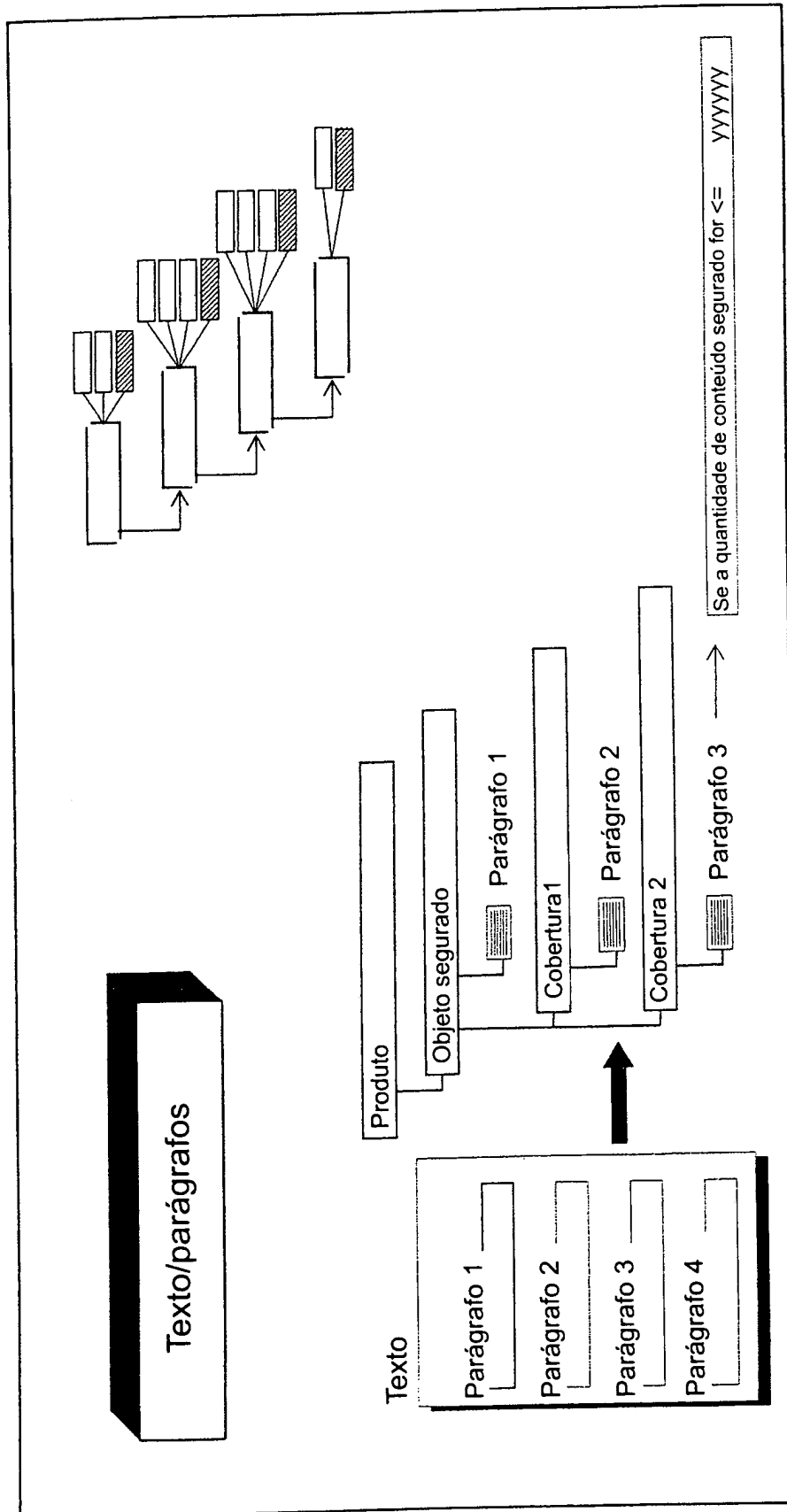
MOVE

FIG. 18b

Sesión A - [32 x 80]		Archivo Edición Vista Comunicación Acciones Ventana Ayuda	
File	Edit	Confirm	Menu Utilities Compilers Test Help
VIEW	MAIS.LIBEPRO.COPYS(TP3CCYAU) - 01.00	Columns 00001 00072	Scroll ==> CSR
Command ==>			
032612	COMPUTE WSC-00000001092-00000006092-01 =		
032613	WSR-00000006178-00		
032614	0,1		
032615	ON SIZE ERROR		
032616	PERFORM CALCULT-ERROR THRU CALCULT-ERROR-EXIT		
032617	END-COMPUTE.		
032618	MOVE 'CCL' TO TR-EX-TYPE OF TR-EX-VALUES		
032619	PERFORM TR00001092-000006092-0100001		
032620	THRU TR00001092-000006092-0100001-E.		
032621	0000000109200000060920000101-X.		
032622	EXIT.		
032623	0000000109200000060920000105 .		
032624	INITIALIZE TR-EX-VALUES		
032625	REPLACING ALPHANUMERIC DATA BY SPACES		
032626	NUMERIC		
032627	COMPUTE WSC-00000001092-00000006092-05 =		
032628	250000		
032629	ON SIZE ERROR		
032630	PERFORM CALCULT-ERROR THRU CALCULT-ERROR-EXIT		
032631	END-COMPUTE.		
032632	MOVE 'CCL' TO TR-EX-TYPE OF TR-EX-VALUES		
032633	PERFORM TR00001092-000006092-0500001		
032634	THRU TR00001092-000006092-0500001-E.		
032635	0000000109200000060920000105-X.		
032636	EXIT.		
032637	F2=Split F3=Exit F5=Rfind F6=Rchange F7=Up		
F1=Help	F10=Left F11=Right F12=Cancel		
F8=Down			
MA a		04/015	



FIG. 19



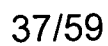


FIG. 20b

Sesión A - [32x80]
Archivo Edición Vista Comunicación Acciones Ventana Ayuda

Process Options Utilities Help

FM/DB2 (DBA1)

Command ==>

TABLE AISDV1.KTPAGST

NUVERPAR CTS0A65T CDTEXT0

N8 N9 N10

DEC(3,0) DEC(5,0) CH(8)

<UN> <UN-> <UN+-->

**** Top of data ****

0 1 TDAPPGEN

0 1 TDQUOGEN

**** End of data ****

Table Browse

0 of 2

Scroll CSR

Format IABL

TCROLD

N15

CH(3)

<->

F1=Help

F2=Zoom

F3=Exit

F4=CRetriev

F5=RFind

F6=RChange

F7=Up

F8=Down

F9=Swap

F10=Left

F11=Right

F12=Cancel

MA

a

04/015

38/59

FIG. 21a

Sesión A: [32 x 80]
Archivo Editar Yoda Comunicación Archivos Ventanas Ayuda

File Edit Confirm Menu Utilities Compilers Test Help

VIEW
Command ==>
000320
000321
000322
000323
000324
000325
000326
000327
000328
000329
000330
000331
000332
000333
000334
000335
000336
000337
000338
000339
000340
000341
000342
000343
000344
000345
F1=Help
F8=Down

MAIS.LIBEPRO.CO80L(TPTLMD8D) - 01.00
CHARS 'KTPA65T' found
Scroll ==> CSR

EXEC SQL
DECLARE **KTPA65T** CURSOR FOR
SELECT CDTEXT0, TCTIMOV, CDMOTIVO, CDPARRAF,
CTSOA65T, NUVERTXT, NUVERPAR, TCROLDEP,
TCROLDDES, INOBLIGA
FROM **KTPA65T**
WHERE CDPRODTE = :WS-CDPRODTE-P0 AND
CDPRODCO = :WS-CDPRODCO-P0 AND
CDOBJTP = :WS-OBJECT-TX AND
CDMCT = :WS-MCT-TX AND
TCTIMOV = :WS-TRANSACTION-CURRENT AND
CDMOTIVO = :WS-MOTIVE AND
CDCLTEXT = :WS-CLASS AND
CDPARRAF <> "00000000" AND
FEINIGST <= :WS-DATE AND
FEFINIGST >= :WS-DATE
FOR READ ONLY
END-EXEC.

1200-DECLARE-CURSORS-EXIT.

*AIS-PFE-OS-BEGIN

* INCLUDE COMMON PERMANENT ROUTINES FOR TEXTS FUNCTIONALITY *

* EXEC SQL
F2=Split F3=Exit F5=Rfind F6=Rchange F7=Up
F9=Swap F10=Left F11=Right F12=Cancel

MD a

06/033

FIG. 21b

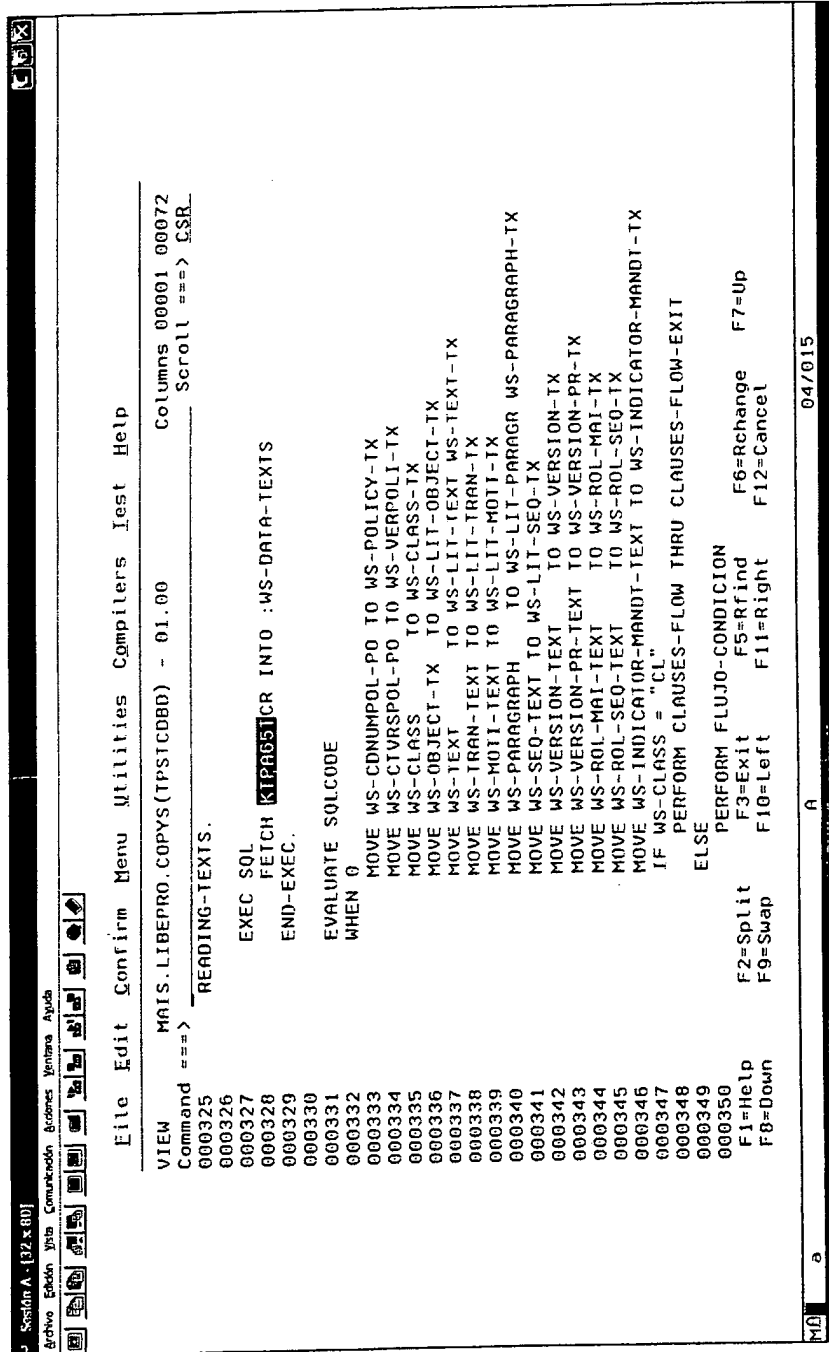


FIG. 22

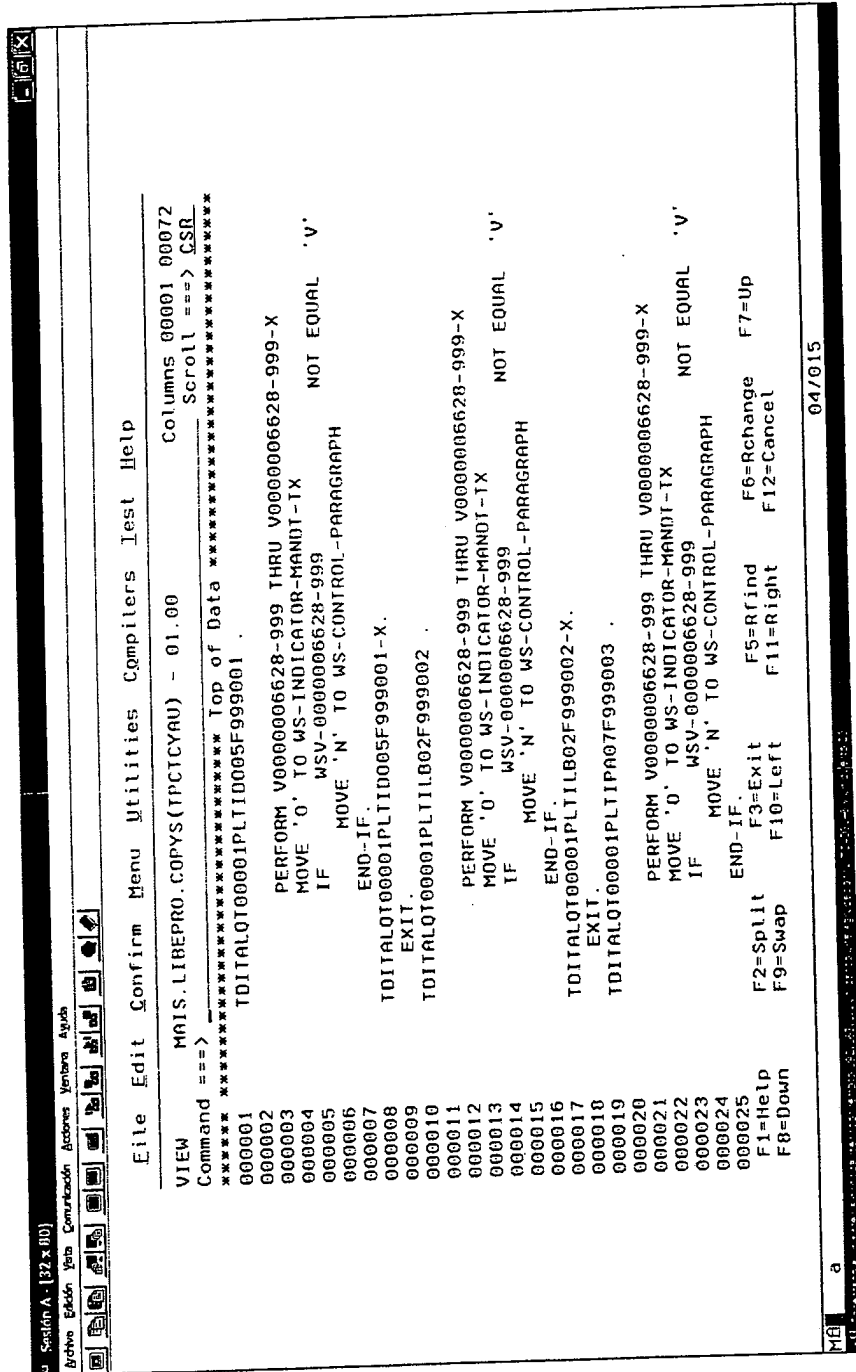


FIG. 23

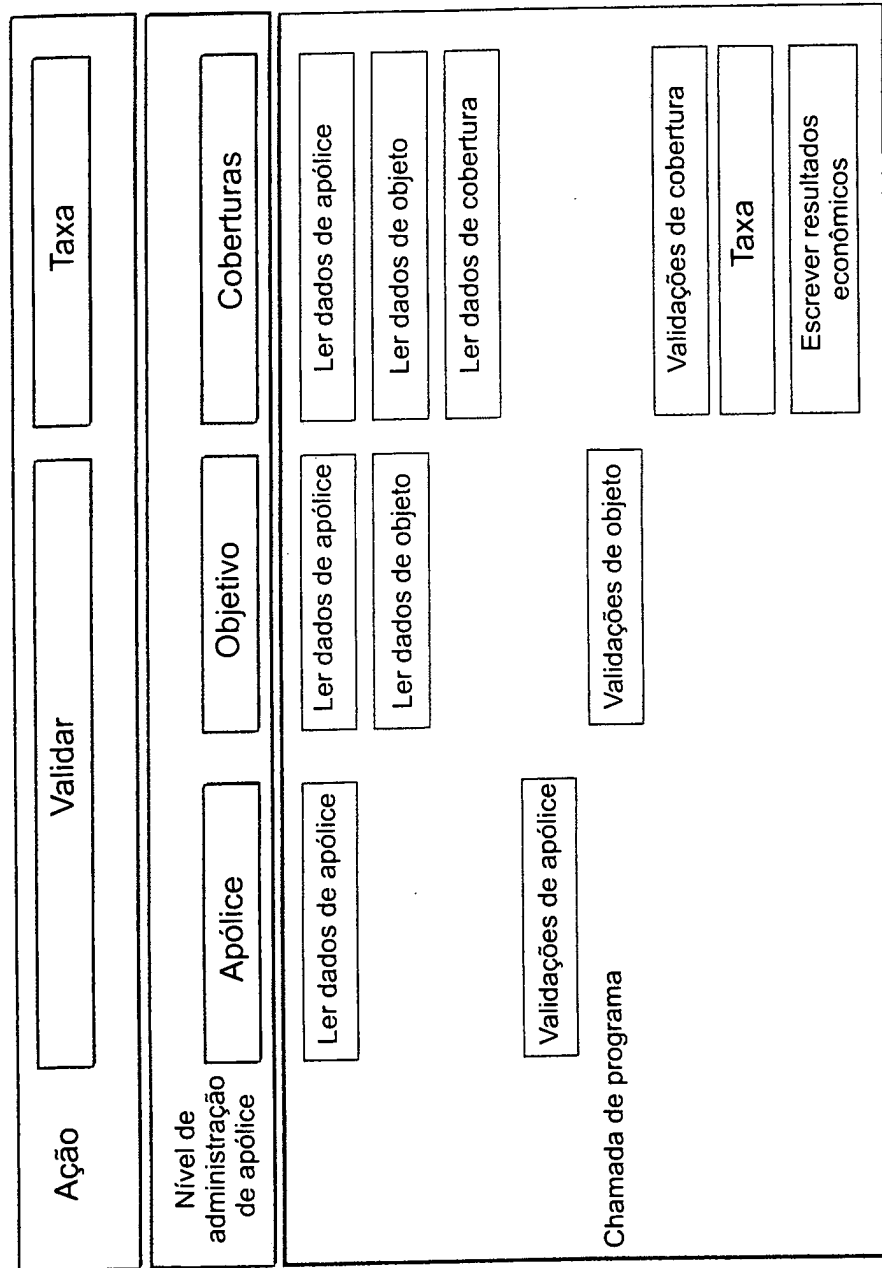


FIG. 24

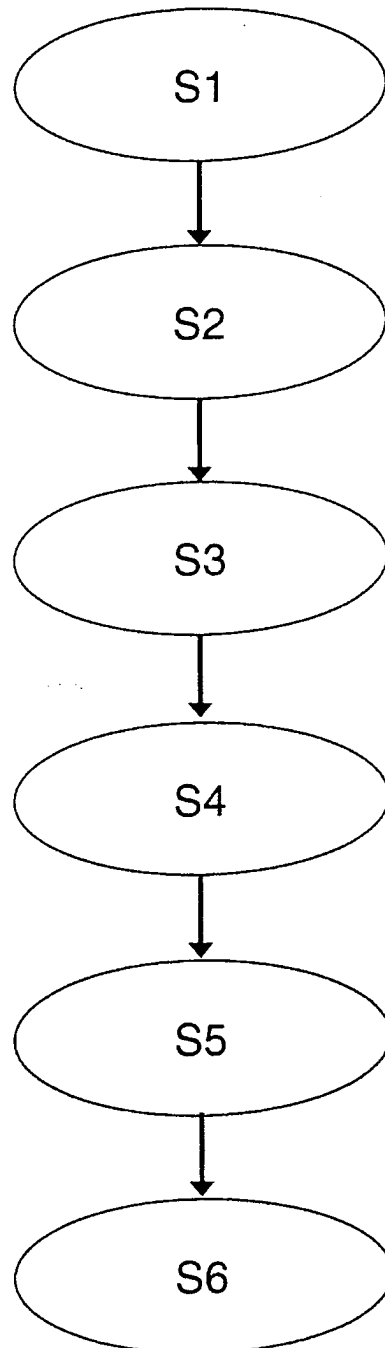


FIG. 25

LOCAL

Development

MASTER Environment

Central Administration

Party & Agent

Policy Admin

Customer Accounting

Central Admin

Window

Exit

Products

User conf

Tables

Business Docs

Task & Notes

Menus

Errors

System adm

Tools

Help

Product search criteria

Template

Street

Product Name

Description

Type

Structure

Description

Relationship type

Status

Insurance type

Search

Add

View

Modify

Delete...

Activate

Deactivate

Inactivate

Close

Template	Street	Product Name	Description	Type	Status	Ins. type
0	000000000000	0	0	TECHNICAL	DEVELOPM...	ACCIDENTS
0000.JR	000000000000	PRUEBA	PRUEBA	TECHNICAL	DEVELOPM...	ACCIDENTS
0000.JR2	000000000000	PRUEBA2	P2	TECHNICAL	DEVELOPM...	AUTOS
0003.JR	000000000000	0003.JR	PRUEBA	TECHNICAL	DEVELOPM...	AUTOS
CORE_MOTOR	000000000000	CORE_MOTOR	CORE_MOTOR	TECHNICAL	DEVELOPM...	AUTOS
MOTORFLEET	000000000000	MOTOR_FLEET	MOTOR_FLEET	TECHNICAL	ACTIVE	AUTOS
MOTORFLEET	STD_FLEET	STREET	MOTOR_FLEET	COMMERCIAL	ACTIVE	AUTOS
YACHT	000000000000	YACHT PROTECTOR	YACHT PROTECTOR	TECHNICAL	DEVELOPM...	PENSIONS

Refresh

Design...

16

user name

23/02/2010

02:12 PM

FIG. 27a

Functional Integration 2 - Central Administration

Party & Agent

Policy Admin

Customer Accounting

Central Admin

Reinsurance

Window

Exit

Products

User Conf

Tables

Business Docs

Task & Notes

Menus

Errors

System Adm

Tools

Help

Central Administration / Products / Coherency Check Window

Template

Street

COMBINED

SBC_BUS

Execute Coherency Check

☒ Elements coherency
 ☒ Validations coherency
 ☒ Ratings coherency

Validation code	Object	Guarantee	Validation type	Item code 1	Item name 1
PRODUCT STRUCTURE COHERENCY			E		
PRODUCT STRUCTURE COHERENCY			E		
ELEMENTS DEFINED ON PF			E		
ELEMENTS DEFINED ON PF			E		
ELEMENTS DEFINED ON DO			E		
ELEMENTS DEFINED ON DO			E		
DRILL DOWN ELEMENTS DEF.			E		
DRILL DOWN ELEMENTS DEF.			E		
PROTECTED ELEMENTS DEF.			E		
PROTECTED ELEMENTS DEF.			E		
RESTRICTIONS TABLES DEF.			E		
RESTRICTIONS TABLES DEF.			E		
STREET CODE DEFINITION			E		
STREET CODE DEFINITION			E		
ELEMENTS LENGTH VALIDATION			E		
ELEMENTS LENGTH VALIDATION			E		

Close

19/11/2007

User name

04:00 PM

FIG. 27b

Functional Integration 2 - Central Administration

Party & Agent

Policy Admin

Customer Accounting

Central Admin

Reinsurance

Window Exit

Products

User Conf

Tables

Business Docs

Task & Notes

Menus

Errors

System Admin

Tools

Help

Central Administration / Products / Coherency Check Window

Template

Street

COMBINED

SBC_BUS

Execute Coherency Check

☒ Elements coherency
 ☒ Validations coherency
 ☒ Ratings coherency

Validation code	Object	Guarantee	Validation type	Item code 1	Item name 1
OPERATION ELEMENTS DEFINED			R	VANETRAT	NET RATE EXP
OPERATION ELEMENTS DEFINED			R	VANETRAT	NET RATE EXP
OPERATION ELEMENTS DEFINED			R	VANETRAT	NET RATE EXP
OPERATION ELEMENTS DEFINED			R	VANETRAT	NET RATE EXP
OPERATION ELEMENTS DEFINED			R	VANETRAT	NET RATE EXP
OPERATION ELEMENTS DEFINED			R	VABOOKBR	VABOOKBR
OPERATION ELEMENTS DEFINED			R	VABOOKBR	VABOOKBR
OPERATION ELEMENTS DEFINED			R	VABOOKBR	VABOOKBR
OPERATION ELEMENTS ON DO			R	VABOOKBR	VABOOKBR
OPERATIONS NOT DEFINED TWICE			R		AD COV COR...
OPERATIONS NOT DEFINED TWICE	Additional Cov ...	Fidelity	R	0000031579	FIDELITY COR...
OPERATIONS NOT DEFINED TWICE	Additional Cov ...	Fidelity	R	0000031577	FIDELITY COR...
OPERATIONS NOT DEFINED TWICE	Trade Bl ER	DI ER - Primer	R	0000000000	DI ER - Primer

user name

19/1/2007

04:00 PM

FIG. 27C

Functional Integration 2 - Central Administration

Party & Agent

Policy Admin

Customer Accounting

Central Admin

Reinsurance

Window

Exit

Products

User Conf

Tables

Business Docs

Task & Notes

Menus

Errors

System

Adm

Tools

Help

Central Administration / Products / Coherency Check Window

Template

COMBINED

Street

SBC_BUS

Execute Coherency Check

☒ Elements coherency
 ☒ Validations coherency
 ☒ Ratings coherency

Check

Validation code

UPCKA IJUN ELEMENTS DEFINED

OPERATION ELEMENTS ON DD

OPERATION ELEMENTS ON DD

OPERATIONS NOT DEFINED TWICE

OPERATIONS NOT DEFINED TWICE

OPERATIONS NOT DEFINED TWICE

OPERATIONS NOT DEFINED TWICE

OPERATIONS NOT DEFINED TWICE

OPERATIONS NOT DEFINED TWICE

OPERATIONS NOT DEFINED TWICE

OPERATIONS NOT DEFINED TWICE

VALIDATIONS CONNECTED

VALIDATIONS CONNECTED

VALIDATION HAS FIRST ELEMENT

VALIDATION HAS FIRST ELEMENT

Object

Additional Cov ... Fidelity

Additional Cov ... Fidelity

Trade PL/PR

Trade PL/PR

Clinical Trial Tr ...

Clinical Trial Tr ...

Trade EL

Trade EL

Guarantee

Fidelity

Fidelity

PL/PR - Proper ...

PL/PR - Proper ...

Clinical Trial - I ...

Clinical Trial - I ...

EL - Injury

EL - Injury

Validation type

R

R

R

R

R

R

R

R

Item code 1

VADOUNAR

Item name 1

VADOUNAR

Close

user name

19/11/2007

04:00 PM

FIG. 29

DESARROLLO - ~ - Contracting

110 {

112 {

126

Add project

Prop./mod. Qu.

Policy cancl.

Receipt que.

Add policy

Pend. policy M.

Claim query

Policy que.

People query

~New policy~

~Policy~

99000333286128

~Vers.~

0

~Tech.cod.~

0

~Comm.cod.~

00001

~Product name~

UTX PROD & UW COM 1

~Additional policy data~

~Data~

Item	COMPANY NAME	TRADE CODE
1184		1204
1185		1205

✓ ~Accept~

X ~Cancel~

01

10MASWA

31/03/2003

13.02

FIG. 30

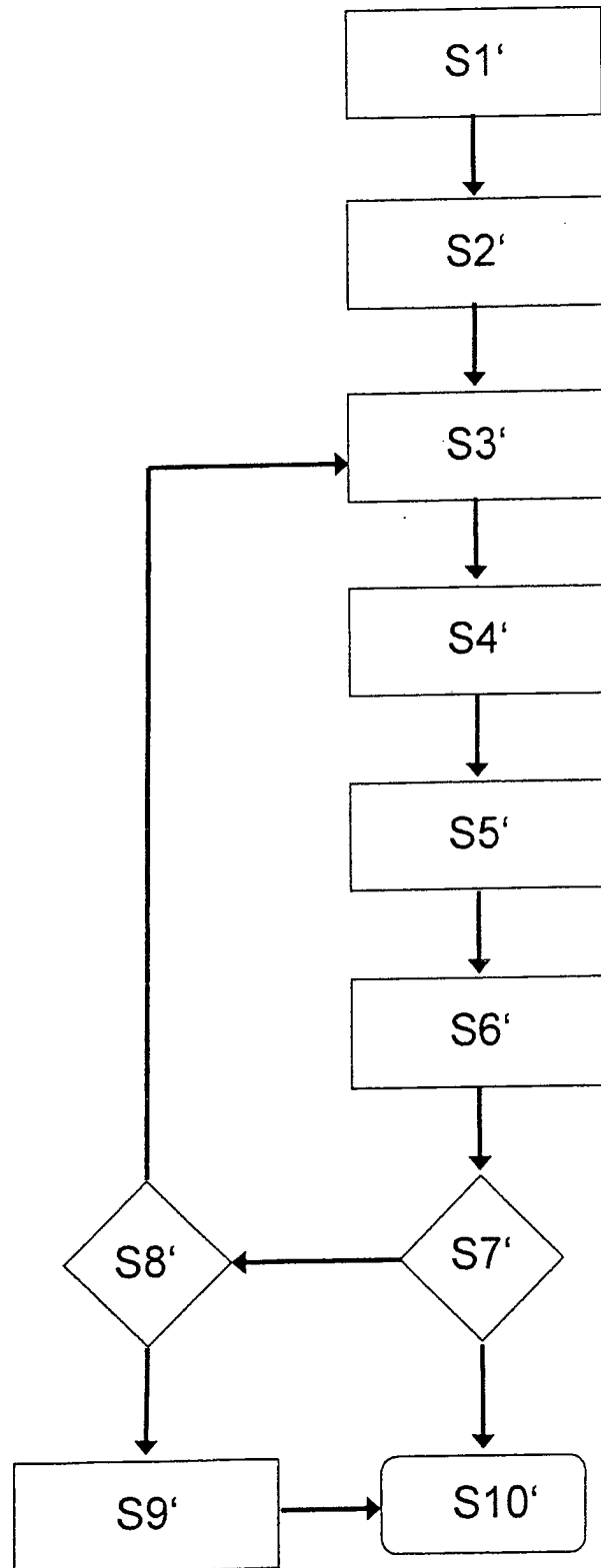


FIG. 31

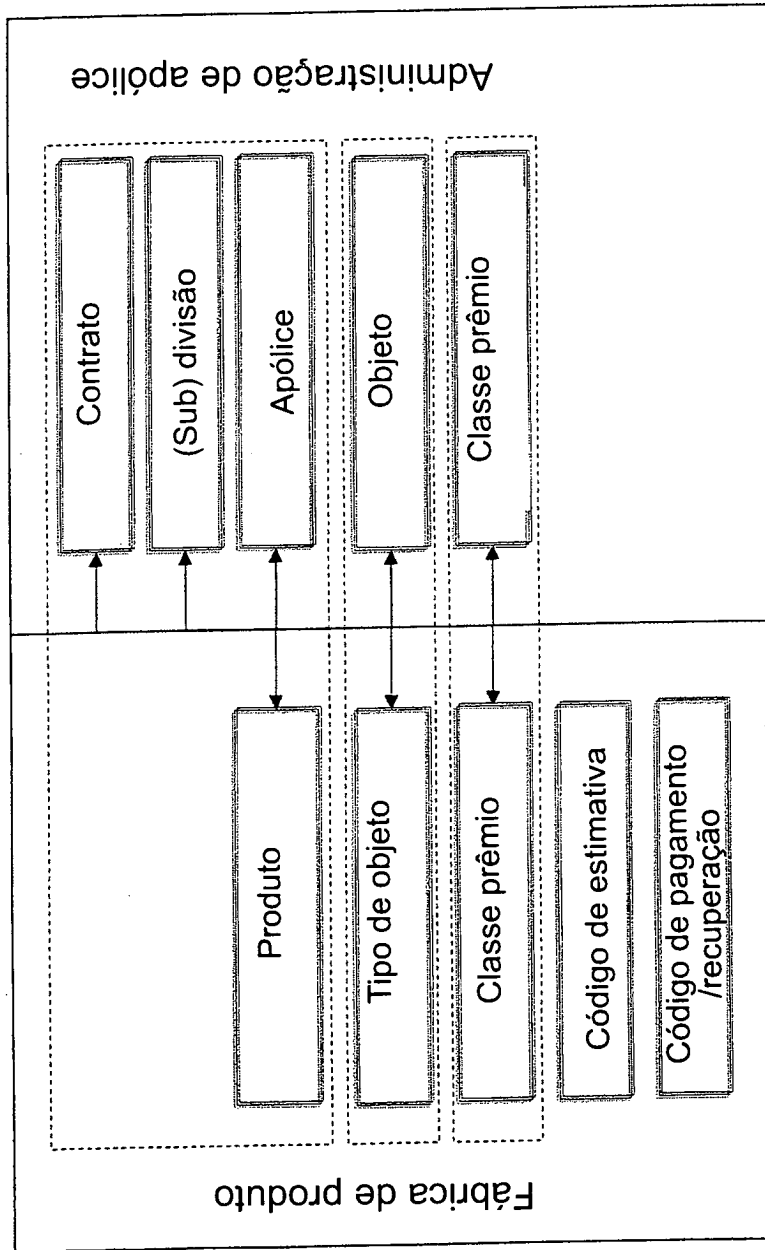


FIG. 32

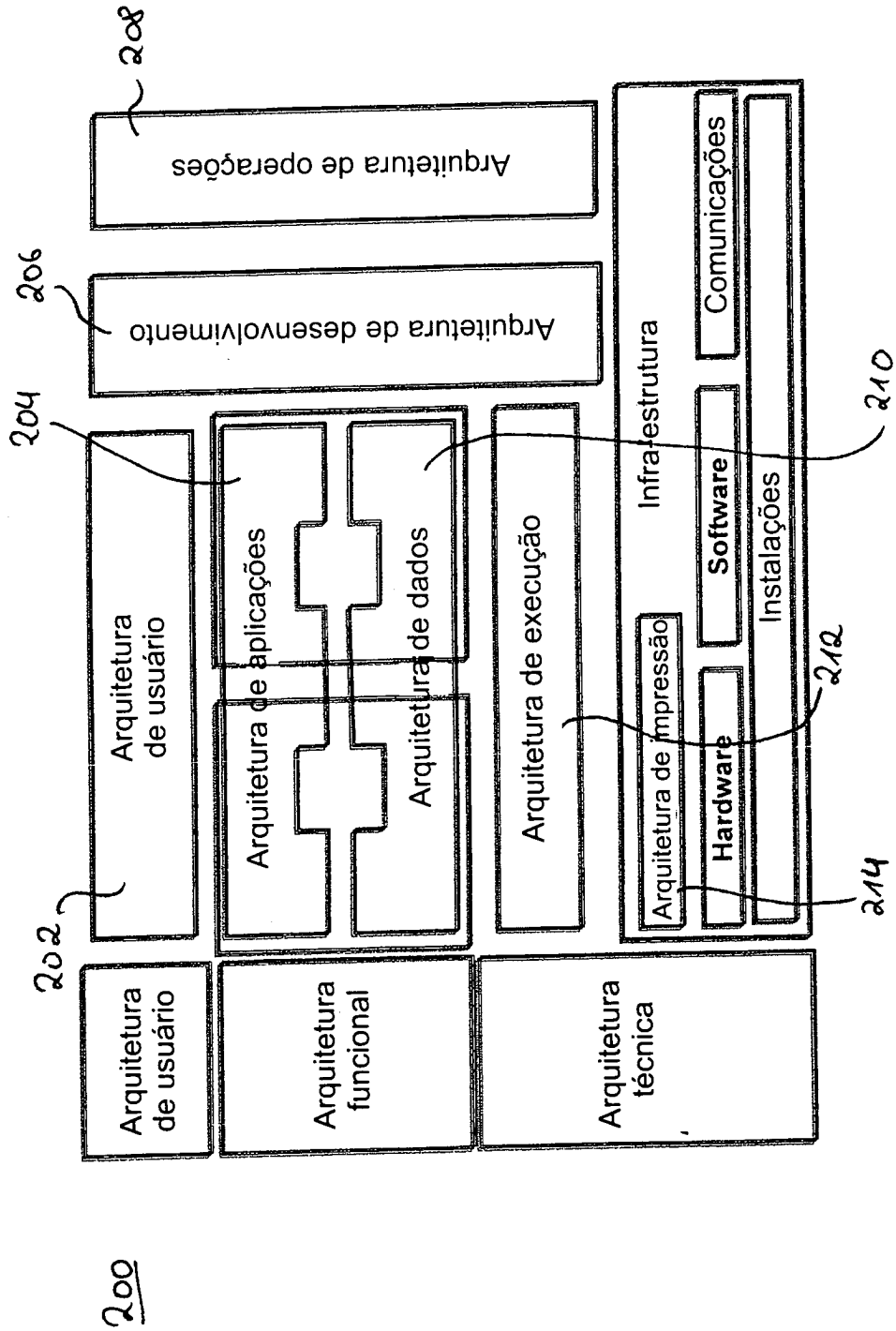


FIG. 34

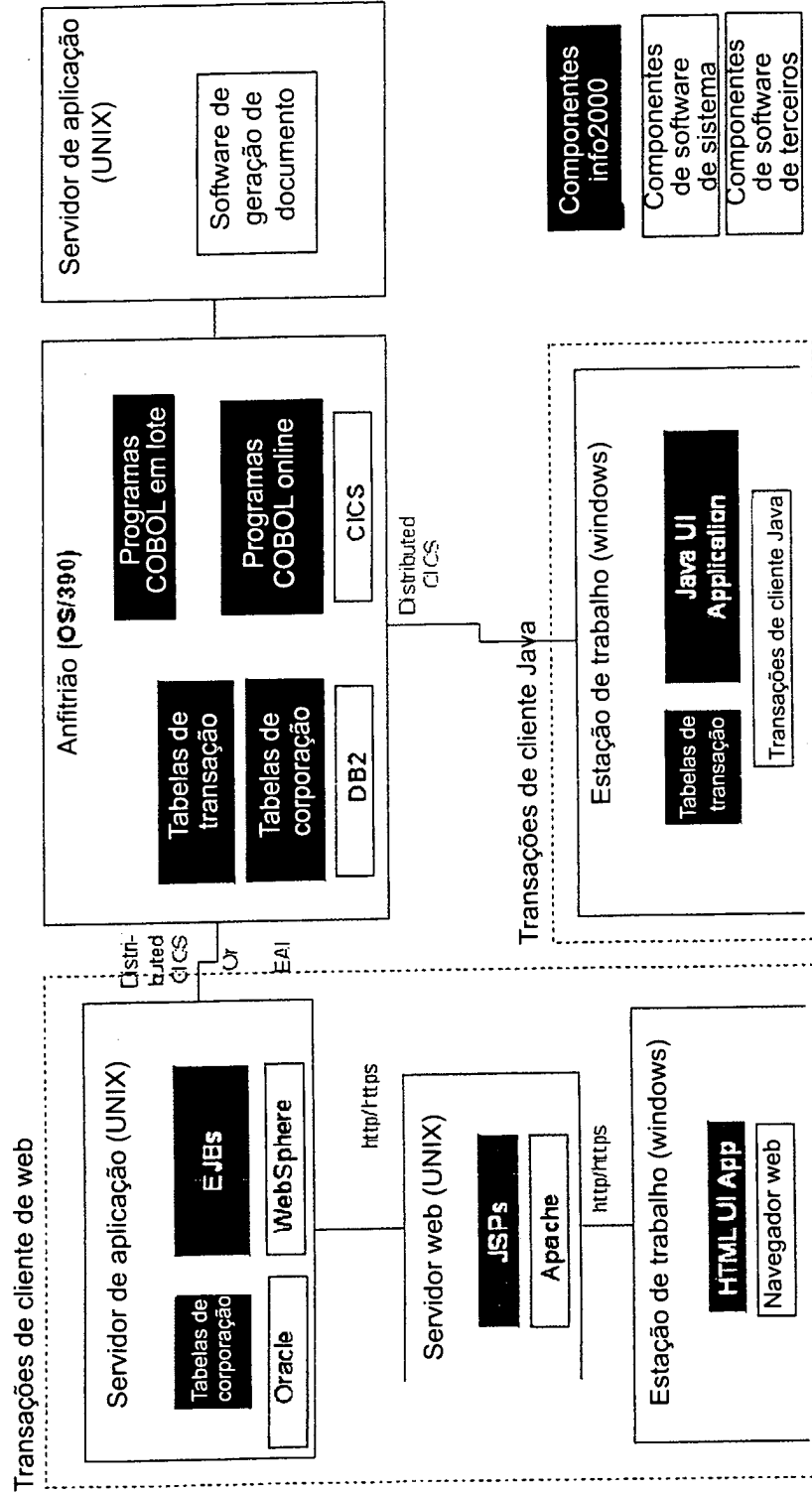


FIG. 35

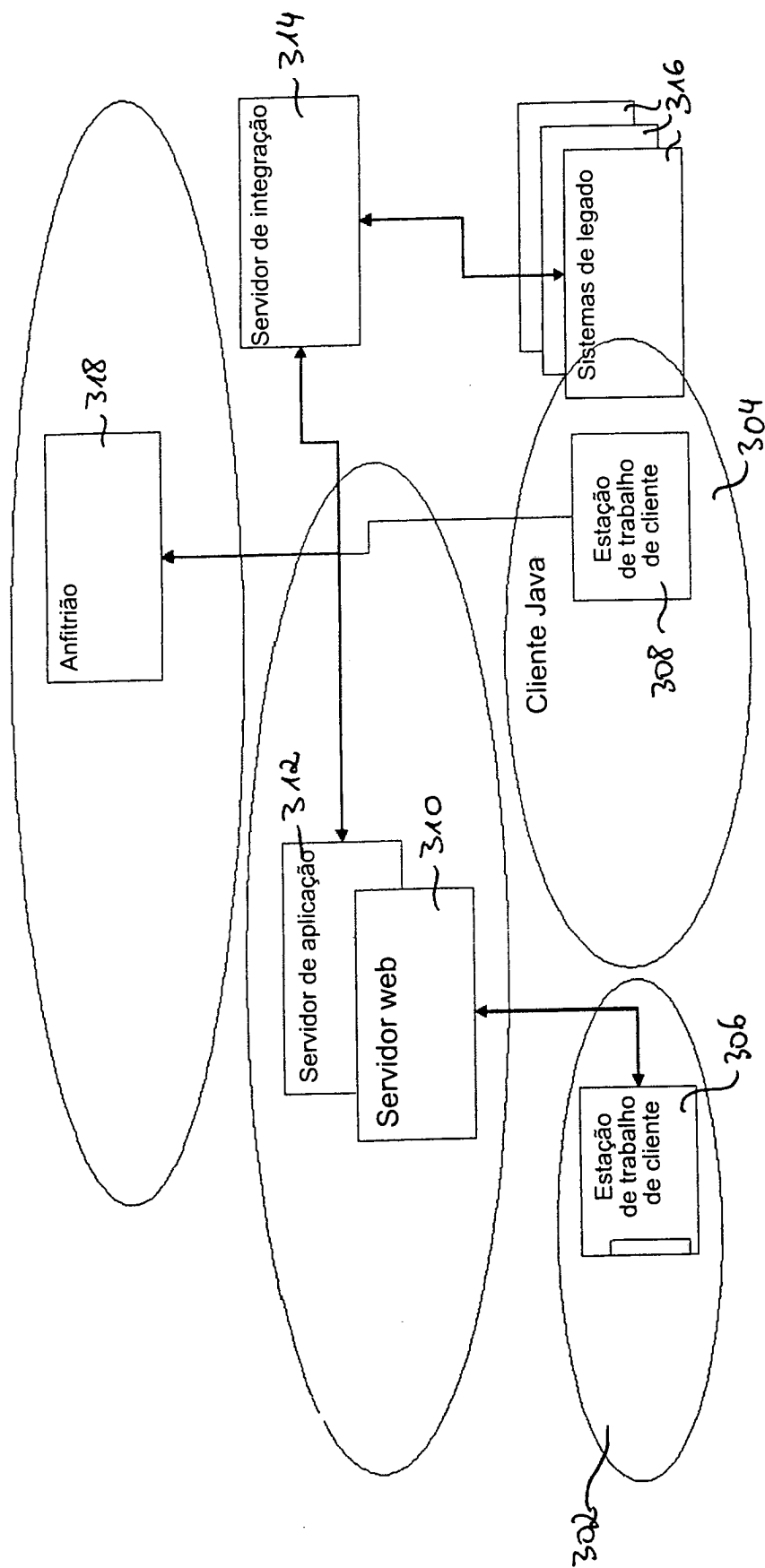
300

FIG. 36

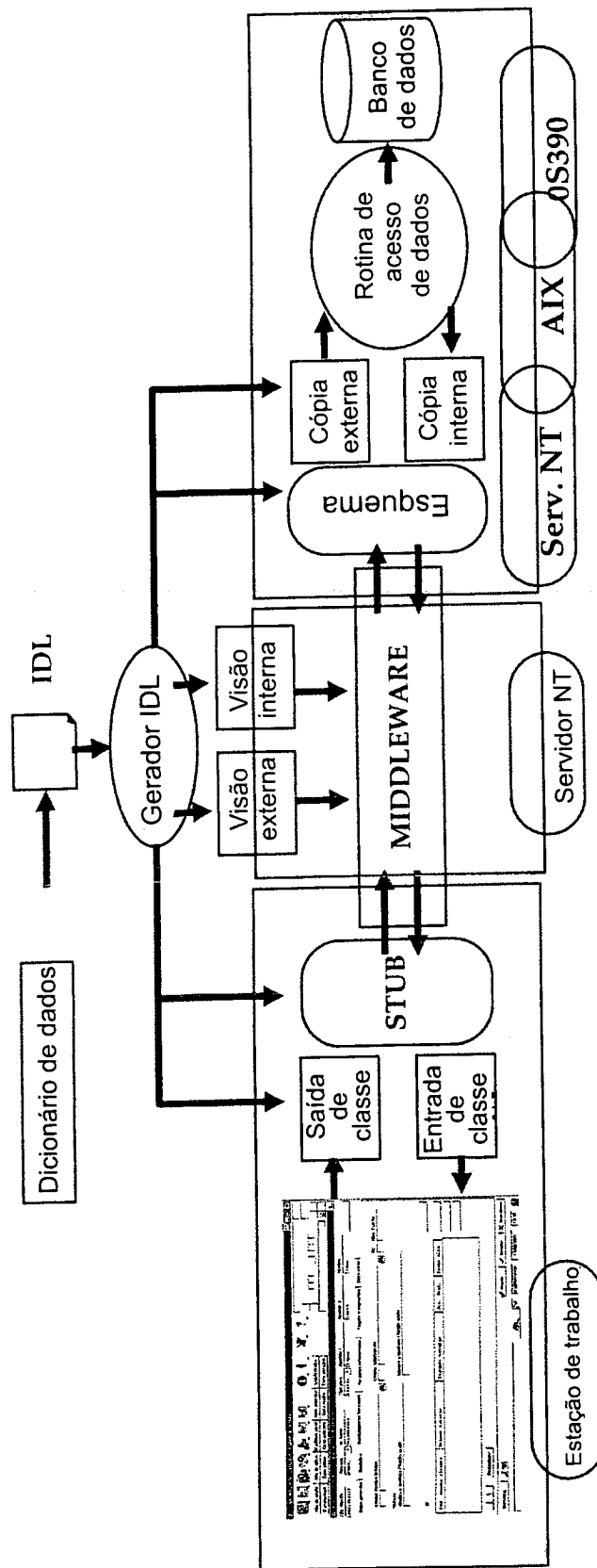
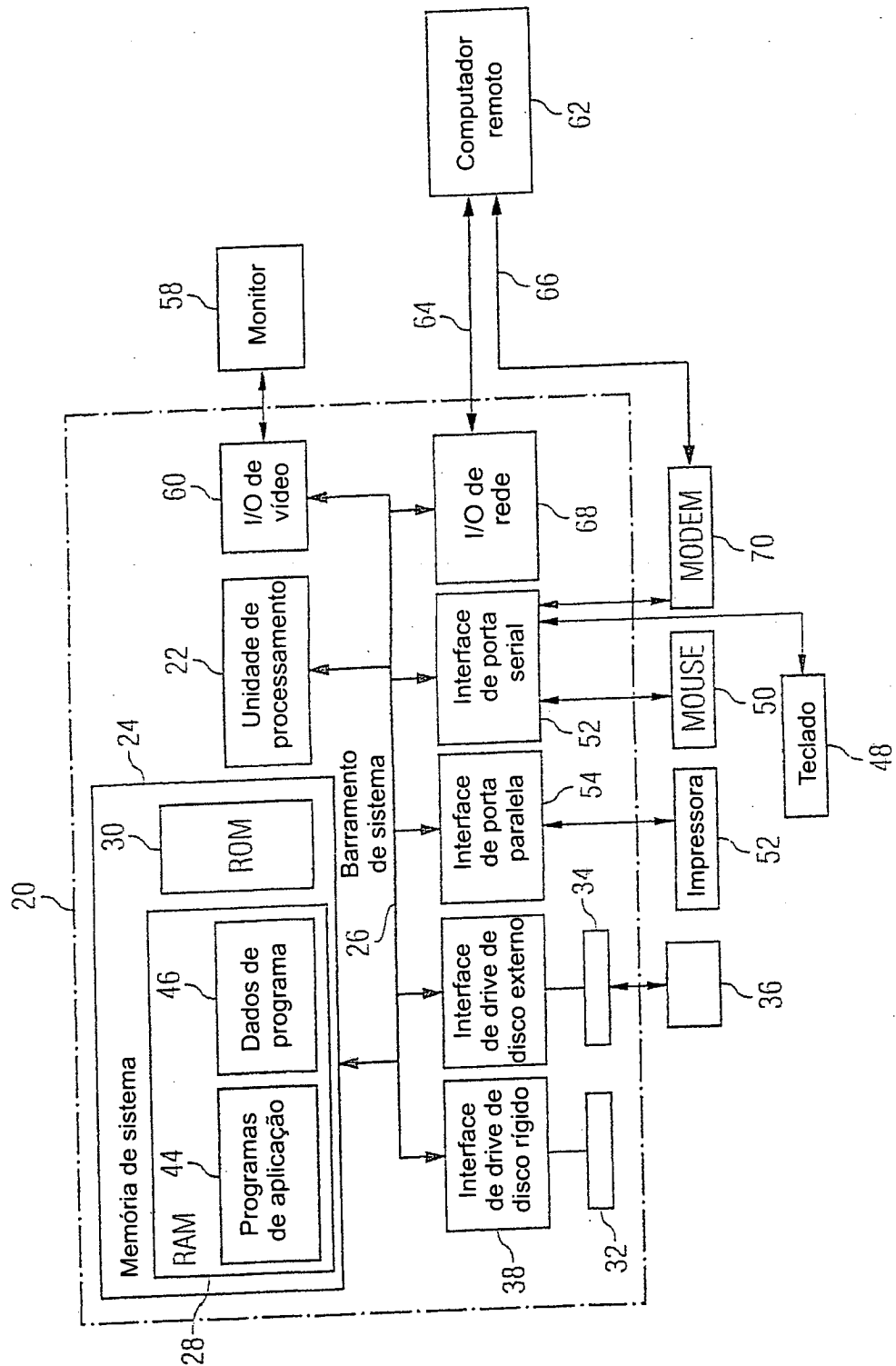


FIG. 37



RESUMO

Patente de Invenção: **"MÉTODO IMPLEMENTADO EM COMPUTADOR PARA ADAPTAR E USAR UM AMBIENTE DE COMPUTADOR, PRODUTO PROGRAMA DE COMPUTADOR E SISTEMA BASEADO EM COMPUTADOR"**.

A presente invenção refere-se a um método implementado por computador para adaptar e utilizar um ambiente de computação, em que o ambiente de computação compreende

um conjunto de arquivos executáveis,

uma estrutura repositória,

uma estrutura de tabela, em que a estrutura repositória e a estrutura de tabela são associadas uma à outra, e

uma interface gráfica de usuário (GUI 101) que é adaptada para iniciar

a execução dos arquivos executáveis,

a leitura de dados e/ou escrita de dados na estrutura de tabela, e proporcionar o dado de saída como texto legível por humano,

o método implementado por computador que tem as etapas de:

proporcionar uma definição de uma alteração do ambiente de

computação em um lado de cliente,

adaptar os arquivos executáveis, de acordo com a definição da alteração do ambiente de computação em um lado de servidor,

introduzir pelo menos um elemento repositório adicional na estrutura repositória no lado de servidor, em que pelo menos um elemento repositório adicional corresponde à adaptação dos arquivos executáveis,

adaptar a estrutura de tabela no lado de servidor, que corresponde ao elemento repositório adicional da estrutura repositória, e

acessar a estrutura de tabela adaptada no lado de servidor através da dita GUI (101) e/ou executar um ou mais arquivos executáveis no lado de servidor através da dita GUI (101), enquanto mantém substancialmente o esboço visual da dita GUI (101), assim como, um produto de programa de computador e um sistema de computador.