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

(11) PI 0513989-9 B1

(22) Data do Depósito: 02/08/2005

(45) Data de Concessão: 21/05/2019

(54) Título: MÉTODO PARA ATUALIZAR SOFTWARE RESIDENTE EM DIFERENTES APARELHOS E APARELHOS ADAPTADOS PARA SEREM ATUALIZADOS PELO MESMO

(51) Int.Cl.: H04N 21/236; H04N 21/434; H04N 21/443; H04N 21/458.

(52) CPC: H04N 21/23614; H04N 21/4348; H04N 21/4432; H04N 21/4586.

(30) Prioridade Unionista: 04/08/2004 FR 0451779.

(73) Titular(es): THOMSON LICENSING.

(72) Inventor(es): FRÉDÉRIQUE ALBERT; HAMID BOUSSAAD; ERIC GAUTIER; JEAN-LUC JUMPERTZ; STÉPHANE RABOISSON.

(86) Pedido PCT: PCT EP2005053768 de 02/08/2005

(87) Publicação PCT: WO 2006/013204 de 09/02/2006

(85) Data do Início da Fase Nacional: 30/01/2007

(57) Resumo: MÉTODO PARA ATUALIZAR SOFTWARE RESIDENTE EM DIFERENTES APARELHOS E APARELHOS ADAPTADOS PARA SEREM ATUALIZADOS PELO MESMO A invenção diz respeito a um método que permite que diferentes versões do software endereçadas a diferentes decodificadores sejam misturadas em um fluxo contínuo comum. O princípio consiste em definir um formato de dados exclusivo que permite que o decodificador seja atualizado a partir de um fluxo contínuo que transporta os segmentos binários de diversas versões de software do software residente, os ditos segmentos binários consistindo em segmentos comuns a diversas versões de software do software residente, os ditos segmentos binários consistindo em segmentos comuns a diversas versões do software e segmentos específicos a cada uma das versões do software.

"MÉTODO PARA ATUALIZAR SOFTWARE RESIDENTE EM DIFERENTES APARELHOS E APARELHOS ADAPTADOS PARA SEREM ATUALIZADOS PELO MESMO"

A presente invenção diz respeito à atualização do software residente contido nos dispositivos. O contexto é mais particularmente a atualização pela distribuição em um fluxo do software de gerenciamento do dispositivo.

Particularmente no domínio dos dispositivos de recepção de televisão digital, comumente chamados decodificadores de televisão digital, é de conhecimento distribuir versões de atualização do software residente no decodificador em fluxos do tipo de transporte contínuo MPEG (Sistema MPEG-2: ISO/IEC, 1994. Codificação Genérica de Imagem em Movimento e Áudio Associado: Sistemas, (Especificação de Sistemas MPEG-2), novembro, ISO/IEC 13818-1). Entende-se que software residente significa todo o software embutido em um dispositivo usado para fazer com que o dito dispositivo opere. Neste domínio, uma dada versão do software residente é, em geral, aplicada a um decodificador de acordo, entre outras coisas, com a versão do hardware deste decodificador, a operadora que gerencia o decodificador ou mesmo o uso planejado para este decodificador. É usual construir um fluxo MPEG, chamado fluxo contínuo de transferência, que contém uma versão do software residente projetada para atualizar uma versão bem definida do decodificador.

No início da implementação dos decodificadores digitais, cada fluxo de atualização transportava todos os segmentos que constituíam o software funcional e era transmiti-

do a um tipo particular de plataforma. À medida em que o número de assinantes aumentou, as operadoras freqüentemente consultaram novos fabricantes para diversificar seus fornecedores de decodificadores. Estes fornecedores procuraram eles mesmos reduzir o custo do equipamento fornecendo novos decodificadores baseados em novas versões de hardware. O equipamento de hardware então tornou-se heterogêneo e a necessidade por largura de banda alocada para os fluxos contínuos de transferência tornou-se maior.

Ao mesmo tempo, para diferentes versões de hardware dos decodificadores que um fabricante fornece para uma operadora, as correspondentes versões do software muitas vezes compartilham segmentos de código comuns. Estes segmentos são, em geral, localizados mais particularmente nas camadas superiores da arquitetura de software (pilhas de protocolo, aplicativos, interface, etc.).

Um problema idêntico é encontrado com fabricantes que usam transferência de software nas fases de produção. De fato, em um dado momento, eles podem ser solicitados para fabricar decodificadores para diferentes operadoras na mesma linha de produção e da mesma versão de equipamento de hardware, sendo os ditos decodificadores transferidos com diferentes versões do software, as ditas versões do software compartilhando segmentos de código comuns. Estes segmentos são, em geral, localizados mais particularmente nas camadas inferiores da arquitetura de software (sistema operacional, acionadores, etc.).

Portanto, o problema técnico envolve a redução das

exigências de largura de banda alocada para a fluxo contínuo de transferência, e isto é tanto para a operadora quanto para o fabricante. Este problema pode ser expresso da seguinte maneira:

Como pode uma operadora reduzir a taxa de bits alocada para os fluxos que transportam diferentes versões do software dedicadas a diferentes plataformas, quando estas diferentes versões do software compartilham segmentos de códigos comuns (pilhas de protocolos, aplicativos, interface, etc.)?

Como pode um fabricante melhorar seu processo de fabricação reduzindo a taxa de bits alocada para os fluxos contínuos de transferência transmitidos nas suas instalações de produção, quando esses fluxos são usados para transferir diferentes versões do software dedicadas a diferentes operadoras em uma única plataforma, as ditas diferentes versões do software tendo segmentos de código comuns (sistema operacional, acionadores, etc.)?

Como um corolário destes problemas, é possível acrescentar como o tempo de descarregamento pode ser reduzido para melhorar a experiência do usuário?

A invenção permite que diferentes versões do software projetadas para diferentes decodificadores sejam misturadas no mesmo fluxo. O princípio é definir um formato de dados exclusivo que permite que um decodificador seja atualizado a partir de um fluxo que transporta os segmentos binários de diversas versões do software residente, sendo estes segmentos binários compostos de segmentos comuns a di-

versas versões do software e segmentos específicos a cada uma das versões do software.

Como um corolário, o procedimento implementado pelo carregador para recuperar os segmentos de código dedicados a uma versão de hardware do decodificador é mais complexo. No entanto, este procedimento pode ser melhorado de maneira tal que o carregador somente procure recuperar os segmentos de código que devem ser atualizados.

A invenção diz respeito a um método para construir uma mídia de distribuição que contém uma pluralidade de versões do software residente planejadas para uma pluralidade de dispositivos, que compreende uma etapa de identificação nas diferentes versões do software residente que compõem a pluralidade das partes comuns para pelo menos duas versões do software residente, bem como uma etapa de divisão das diferentes versões do software residente que compõem a pluralidade nos segmentos, bem como a montagem destes segmentos em uma mídia de distribuição, sendo os segmentos que compõem uma parte comum a pelo menos duas versões encontrados uma vez na mídia de distribuição.

De acordo com uma modalidade particular, a invenção também compreende uma etapa de marcação de cada segmento por meio da identificação dos dispositivos visados pela(s) versão(s) do(s) software(s) residente(s) da(s) qual(s) o segmento é uma parte.

De acordo com uma modalidade particular da invenção, a pluralidade de dispositivos é uma pluralidade de dispositivos de recepção de televisão digital.

De acordo com uma modalidade particular da invenção, a mídia de distribuição é um fluxo contínuo de transporte MPEG.

A invenção também diz respeito a um método de aquisição de toda ou parte de uma versão do software residente por um dispositivo que tenha meios de acesso para uma mídia de distribuição distribuída por uma rede de distribuição, a dita mídia compreendendo uma pluralidade de segmentos que compõe uma pluralidade de versões do software residente, o método compreendendo uma etapa de identificação na mídia de distribuição de pelo menos um segmento destinado ao dito dispositivo, bem como uma etapa de transferência do dito segmento identificado no dispositivo para a dita mídia de distribuição e o armazenamento na memória do dito segmento.

De acordo com uma modalidade particular da invenção, a etapa de identificação na mídia de distribuição compreende uma etapa de comparação de um conjunto de identificadores que identifica em cada segmento os dispositivos visados pela versão do software residente a qual o segmento pertence e um conjunto de identificadores residentes no dispositivo, e a codificação do seu tipo e uso.

De acordo com uma modalidade particular da invenção, o conjunto de identificadores contém pelo menos um identificador de plataforma que identifica o tipo específico da versão de hardware do dispositivo, um identificador de produto que identifica a operadora que opera o dispositivo e um identificador de uso que identifica o uso planejado do dispositivo.

De acordo com uma modalidade particular da invenção, o dispositivo é um receptor de televisão digital e a mídia de distribuição é um fluxo contínuo de transporte MPEG.

A invenção também diz respeito a uma mídia de distribuição de uma pluralidade de versões do software residente projetadas para uma pluralidade de dispositivos, sendo as ditas versões do software divididas em segmentos onde a dita mídia de distribuição concentra os segmentos da pluralidade de versões do software residente, os segmentos comuns a pelo menos duas versões do software estando presentes somente uma vez na mídia.

De acordo com uma modalidade particular da invenção, cada segmento tem um conjunto de identificadores que identificam em cada segmento os dispositivos visados pela versão do software residente a qual o segmento pertence.

De acordo com uma modalidade particular da invenção, o conjunto de identificadores contém pelo menos um identificador de plataforma que identifica o tipo específico da versão do hardware do dispositivo, um identificador de produto que identifica a operadora que opera o dispositivo e um identificador de uso que identifica o uso planejado do dispositivo.

De acordo com uma modalidade particular da invenção, a mídia tem a forma de um fluxo contínuo de transporte MPEG.

A invenção também diz respeito a um dispositivo que tem meios de acesso para uma mídia de distribuição que

contém uma pluralidade de versões do software residente projetada para uma pluralidade de dispositivos, sendo estas versões divididas em uma pluralidade de segmentos na mídia de distribuição, o dito dispositivo tendo meios para transferir os segmentos presentes na mídia de descrição e meios para armazenar estes segmentos e tendo meios para determinar os segmentos que são parte de uma versão do software residente que foi planejada para ele entre uma pluralidade de segmentos presentes na mídia de distribuição.

De acordo com uma modalidade particular da invenção, o dispositivo tem um conjunto de identificadores que codificam seu tipo e uso, e meios de comparação deste conjunto de identificadores com o conjunto de identificadores presentes em cada segmento presente na mídia de distribuição e que identificam os dispositivos visados pela versão do software residente a qual o segmento pertence.

De acordo com uma modalidade particular da invenção, os meios de acesso a uma mídia de distribuição são meios de acesso a um fluxo contínuo de transporte MPEG.

A invenção será mais bem entendida, e outros recursos e vantagens específicas emergirão da leitura da seguinte descrição, a descrição fazendo referência aos desenhos anexos em que:

A Figura 1 mostra a conhecida arquitetura de um sistema para distribuir software de transferência para um conjunto de decodificadores.

A Figura 2 mostra um exemplo de arquitetura de software de um decodificador.

A Figura 3 mostra os diferentes níveis de divisão do software residente em uma modalidade da invenção.

A Figura 4 mostra o segmento de cabeçalho na modalidade da invenção.

A Figura 5 mostra o formato da seção MPEG na modalidade da invenção.

A Figura 6 mostra as etapas do método usado pelo decodificador para encontrar os segmentos que ele deve transferir na modalidade da invenção.

A Figura 7 mostra a arquitetura de um decodificador de acordo com a modalidade da invenção.

Uma modalidade da invenção será agora descrita. Esta modalidade é localizada no domínio dos decodificadores de televisão digital, mas a invenção pode ser aplicada a qualquer dispositivo que tenha um software residente que possa ser atualizado através de meios gerais de distribuição. Um exemplo que pode ser citado é o de aparelhos de DVD atualizados por um único DVD que contém versões do software destinadas a diferentes plataformas ou outros elementos.

A Figura 7 mostra a arquitetura de um decodificador referenciado 7.1 de acordo com uma modalidade da invenção. Este decodificador é constituído por um sintonizador referenciado 7.7 que permite a recepção dos fluxos que contêm os serviços de áudio e vídeo. Este módulo pode ser um satélite, cabo, módulo terrestre ou até uma interface de conexão de rede do tipo IP. O fluxo deste sintonizador será então processado pelo demultiplexador / decodificador MPEG referenciado 7.6. Este módulo é responsável por verificar os

direitos e decifrar os fluxos, se necessário. O serviço exigido é então isolado no fluxo; de fato, um único fluxo pode conter uma multiplexação de diversos serviços de áudio e vídeo. Uma vez que o serviço exigido é isolado, os diferentes fluxos básicos são extraídos dele. Estes fluxos são, em geral, compostos de um fluxo comprimido de vídeo e um fluxo de áudio, agora, em geral, de acordo com o padrão MPEG-2, mas a invenção opera independente do formato dos serviços, tais como MPEG-4 ou outros. Portanto, estes fluxos são descomprimidos e então convertidos para sinais analógicos para fornecer uma saída de vídeo e uma saída de áudio. Todas essas operações são controladas por um software residente armazenado em uma maneira distribuída entre a memória somente de leitura referenciada 7.4 e a memória regravável não-volátil referenciada 7.3. Quanto à memória de acesso aleatório referenciada 7.2, ela será usada como uma memória de trabalho para executar o software residente.

A Figura 2 mostra diagramaticamente os diferentes blocos usados que compreendem o software residente presente na modalidade da invenção. O software residente opera no hardware, referenciado 2.8, que foi descrito anteriormente. Ele é composto de duas partes, uma primeira, referenciada 2.5, consiste em um carregador referenciado 2.6 e um conjunto mínimo de acionadores, referenciado 2.7, que permite que este carregador opere. Tipicamente, este carregador e seus acionadores são armazenados em memória não-regravável, não-volátil a fim de assegurar sua integridade ao longo do tempo. De fato, a função deste carregador é permitir a atuali-

zação do software do decodificador e deve estar apto a operar de qualquer maneira independente do estado do software residente armazenado na memória regravável, ele deve não estar apto a ser corrompido.

A outra parte do software residente, referenciada 2.1, consiste em um conjunto completo de acionadores, referenciado 2.4, que permite que o hardware seja gerenciado. Acima destes acionadores está um sistema operacional referenciado 2.3 que oferece um conjunto de funções genéricas que se baseia em acionadores. O aplicativo, referenciado 2.2, compreende o software de alto nível que provê o usuário com as funções do seu decodificador. Ele é composto de uma interface homem-máquina e implementa as funções tais como a mudança de serviço, gerenciamento dos possíveis mecanismos de interatividade, aplicativos interativos e outras funções. É comum que este componente também compreenda um carregador referenciado 2.9, permitindo que este grupo de softwares referenciado 2.1 seja atualizado. Este carregador pode ser atualizado por si mesmo. De fato, quando ele transfere uma nova versão do sistema do software, esta versão pode conter uma nova versão deste carregador que será colocada na memória no lugar do carregador referenciado 2.9. Se ocorrer um problema durante esta atualização, o carregador referenciado 2.6 armazenado na memória não-volátil sempre será capaz de permitir uma nova atualização do software, resolvendo o problema e encontrando uma versão operacional do componente do software referenciado 2.1 e, portanto, do carregador referenciado 2.9 que é parte dele.

A exigência de transferir software para um decodificador pode ter vários motivos. O primeiro motivo é a corrupção do software instalado. De fato, sendo o componente do software referenciado 2.1 armazenado em memória regravável, é sempre possível que ele se torne corrompido. Neste caso, a corrupção é detectada na próxima inicialização do dispositivo, que irá verificar a integridade dos módulos que compreendem o software por um sistema CRC (Verificação de Redundância Cíclica). Se for detectada uma corrupção em um módulo diferente do carregador, este carregador será usado para transferir uma nova versão perfeita do software. É possível tanto atualizar uma nova versão completa do software do sistema como simplesmente módulos detectados como corrompidos. Quando detecta-se que o próprio carregador está corrompido, a atualização cabe ao carregador permanente referenciado 2.6.

Um outro motivo para a transferência é a disponibilidade de uma nova versão do software residente. Esta nova versão é capaz de corrigir erros presentes na versão atual e/ou adicionar novas funções a este software residente. Neste caso, a detecção de uma nova versão pelo decodificador, durante sua fase de inicialização, irá fazer com que, possivelmente sob certas condições, esta nova versão seja transferida pelo carregador referenciado 2.9 e instalada no decodificador.

Em geral, também é possível ocasionar a transferência e instalação de uma dada versão do software. Neste caso, o usuário ou o engenheiro habilitado, depois de des-

travar todo o sistema de proteção, será capaz de especificar a versão do software residente que ele precisa instalar no dispositivo e transferi-la. Isto pode ser feito durante operações de manutenção ou durante a preparação dos dispositivos na fábrica antes da entrega. Desta maneira, é sempre possível configurar um dispositivo com uma versão escolhida do software do sistema de acordo com o uso para o qual o dispositivo é destinado.

A infraestrutura típica de transferência é descrita na Figura 1. A totalidade da versão do software destinado a um dispositivo é constituída por diferentes módulos que formam, entre outras coisas, os acionadores, o sistema operacional e a camada de aplicativos. Estes módulos referenciados 1.1, 1.2 e 1.3 serão montados e misturados em um fluxo contínuo de transporte MPEG em um misturador referenciado 1.4. Este fluxo será transmitido por uma estação de transmissão referenciada 1.5 a um satélite referenciado 1.6 que irá distribuí-lo para os decodificadores referenciados 1.7 a 1.10. A infraestrutura aqui descrita é aquela da distribuição por satélite, mas o diagrama é o mesmo para um outro tipo de distribuição que pode ser cabo, terrestre ou mesmo uma rede interna em uma instalação de fabricação. Neste caso, o fluxo calculado pelo misturador referenciado 1.4 será transmitido por um servidor responsável por fazer o sinal destinado ao decodificador. De uma maneira padrão, a fluxo contínuo de transferência construído irá conter uma versão do software destinada a um dado tipo de decodificador mas, no contexto da invenção, será possível misturar módulos que

pertençam a várias versões do software neste fluxo.

Um certo número de identificadores é usado para marcar as versões do software residente e determinar os decodificadores relevantes. Os seguintes identificadores basicamente serão encontrados e terão um papel decisivo na determinação do processo de transferência:

- O identificador de plataforma, na realidade composto de um agregado de dois identificadores, PT que indica o tipo de plataforma, que corresponde ao modelo do decodificador, e PV que especifica o número de versão no tipo de plataforma. Assim, este identificador permite que um modelo de hardware do decodificador seja precisamente identificado.

- O identificador de produto PR que irá, em geral, identificar a operadora que usa o decodificador. De fato, um mesmo decodificador de hardware não irá usar o mesmo software dependendo da operadora que implementá-lo.

- O identificador de uso US permite que uma mesma plataforma e um mesmo produto diferenciem o uso que será feito da plataforma. Assim, pode-se diferenciar, por exemplo, uma plataforma projetada para um assinante e uma plataforma projetada para teste ou um outro propósito.

Um conjunto destes identificadores é armazenado no decodificador, fornecendo informações do modelo exato da plataforma, da operadora e do uso para o qual o decodificador foi projetado. Os fluxos contínuos de transferência também contêm um conjunto destes identificadores que permite que os decodificadores, para os quais o software contido foi criado, sejam conhecidos. Neste contexto da invenção, cada

segmento terá seu próprio conjunto de parâmetros.

Em uma maneira conhecida, quando um decodificador tem que ser atualizado com uma nova versão do software pelos motivos supradescritos, ele irá usar os parâmetros internos que permitem-no identificar o fluxo que contém a versão do software que é projetada para ele. Assim, ele se conecta a este fluxo e verifica que os identificadores de plataforma, produto e uso contidos no fluxo correspondem a aqueles identificadores que ele possui. Neste caso, ele irá transferir o software contido no fluxo e armazená-lo.

No contexto da modalidade da invenção, o mecanismo de atualização é baseado na transmissão de segmentos binários referenciados 3.3. Estes segmentos correspondem a uma divisão dos despejos de memória, referenciados 3.1, das diferentes versões do software residente que precisam ser misturadas no fluxo. Estes segmentos podem corresponder diretamente aos diferentes módulos que compõem as versões do software residente ou ser o resultado de um outro modo de divisão. Estes segmentos de código são precedidos por um segmento de cabeçalho referenciado 3.2 que descreve todos os segmentos de código distribuídos. Todos os segmentos são transportados por meio de seções MPEG referenciadas 3.4. De fato, os fluxos contínuos de transporte MPEG são compostos de uma sucessão de 188 seções de byte.

O segmento "cabeçalho" ilustrado na Figura 4 é composto de um "cabeçalho", em seguida um laço que descreve cada um dos segmentos de código incluídos no fluxo e um campo de assinatura que autentica este segmento de cabeçalho e

seu conteúdo.

O cabeçalho do segmento "cabeçalho" compreende os seguintes campos:

* "OUI" (Identificador Exclusivo de Organização) especificado pela DVB ("Difusão de Vídeo Digital", o consórcio responsável pela definição dos padrões de difusão digital) identifica exclusivamente cada fabricante e, portanto, cada segmento descrito pelo cabeçalho.

* "Key Index" exigido para autenticar o fluxo quando é usado um algoritmo baseado em chaves públicas. Ele identifica a chave pública a ser usada para autenticar o fluxo.

* "FlowVersionId" permite que uma mudança no conteúdo do fluxo seja conhecida. Esta informação pode ser distribuída na sinalização e assim permite que o decodificador inicie uma operação de transferência cada vez que ele muda.

* "EncryptedSeed, InitialValue" informação exigida para decifrar os segmentos.

As seguintes informações são encontradas em cada segmento:

* "PlatformType, PlatformVersion" identifica a plataforma de hardware sobre a qual o segmento deve ser carregado. Um segmento comum a todas as plataformas será identificado exclusivamente por um par "(PlatformType, PlatformVersion) = any".

* "ProductId" identifica o produto ou cliente. Um segmento comum a todos os produtos será identificado exclusivamente por um "ProductId = any".

* "UsageId" identifica o uso do decodificador no grupo de dispositivo (assinante, teste, etc.). Um segmento comum a todos os usos será identificado exclusivamente por um "UsageId = any".

* "SegmentVersionId" identifica a versão do segmento. Ele é usado para determinar se um segmento já está presente no decodificador e não necessita ser transferido.

* "SegmentType, SegmentId" corresponde respectivamente ao tipo de segmento (sistema operacional, acionadores, interface, etc.) e ao seu identificador no tipo.

* "CrcMemoryType": Tipo de CRC (Verificação de Redundância Cíclica) calculada no segmento armazenado na memória.

* "HashMemoryType": Tipo de dispersão calculada no segmento armazenado na memória. Exigido para verificar a autenticação do fluxo.

* "SegStorageAdress": Endereço de armazenamento do segmento na memória.

* "SegSizeMemory": Tamanho do segmento na memória (segmento não comprimido).

* "CrcFlowType": Tipo de CRC calculado no segmento transportado no fluxo. Exigido para verificar a integridade dos segmentos durante a recepção.

* "CompressionType": Tipo de compressão calculado no segmento transportado no fluxo. Determina se o segmento está ou não comprimido e que algoritmo de compressão foi usado.

* "EncryptionType": Tipo de encriptação no segmen-

to transportado no fluxo. Determina se o segmento está ou não encriptado e que algoritmo de encriptação foi usado.

* "CRCFlow": CRC do segmento transportado no fluxo.

Finalmente, há:

* "Signature": informação que permite que o fluxo seja autenticado.

O formato de transporte é baseado no formato de transporte em seções MPEG mostrado na Figura 5. Ele contém as informações "OUI", "SegmentNb", "FlowVersionId" descritas anteriormente. Esta informação será usada para recuperar os dados transmitidos no fluxo.

Desta maneira, é possível misturar no mesmo fluxo os segmentos que pertencem a diferentes despejos de memória correspondentes a diferentes versões do software residente. Cada segmento tendo seu próprio conjunto de parâmetros PT, PV, PR e US, o destino de cada segmento é definido. Além disso, o fato de atribuir a alguns desses parâmetros o valor "any" não duplica os segmentos para compartilhar diferentes versões do software. De fato, um segmento compartilhado por diferentes versões do software de um dado produto destinado a todas as plataformas de hardware será marcado com PR igual ao valor do produto e (PT, PV) igual a "any" e o mesmo para outros parâmetros. Começa-se proceder assim em diferentes versões do software residente para montar no fluxo, identificando os componentes comuns a pelo menos duas dessas versões. Cada módulo é de fato marcado pela configuração, ou outros meios, com um conjunto de parâmetros PT, PV, PR e US

que especifica seu destino. A análise destes parâmetros identifica os módulos comuns a diversas versões. Então, quando esses módulos são divididos em segmentos, cada segmento será marcado com o conjunto de parâmetros PT, PV, PR e US correspondente a ele, não sendo os segmentos comuns duplicados durante a construção da mídia de distribuição, aqui o fluxo MPEG.

As principais etapas de transferência são ilustradas na Figura 6.

Uma primeira etapa, referenciada E1, consiste em adquirir o segmento de cabeçalho por uma seção que filtra os campos "TableId, SegmentNb=0, OUI, SectionSyntaxVersion, FlowVersionId".

Uma segunda etapa, referenciada E2, consiste em verificar a autenticação deste segmento de cabeçalho graças a sua assinatura.

Uma terceira etapa, referenciada E3, consiste em construir a lista de segmentos correspondente ao decodificador que realiza a transferência por extração sucessiva (veja tabela seguinte):

Os segmentos dedicados à plataforma (PT, PV), produto PR e uso US do decodificador que realiza a transferência.

Os segmentos dedicados à plataforma, produto, em que US=any.

Os segmentos dedicados à plataforma em que (PT, PV)=any. Neste caso, o valor de US é ignorado.

Os segmentos dedicados ao produto, uso em que (PT,

PV)=any.

Os segmentos dedicados ao produto em que (PT, PV)=any, US=any.

Os segmentos genéricos, a saber, em que (PT, PV)=any, PR=any e independente do valor de US.

Plataforma	Produto	Uso	
PT, PV any	PR any	US any	1
		US = any	2
	PR = any	US não levado em conta	3
PT, PV = any	PR any	US any	4
		US = any	5
	PR = any	US não levado em conta	6

Uma quarta etapa, referenciada, E4, consiste, para uma atualização em função da disponibilidade de uma nova versão do software, somente em recuperar desta lista os segmentos para os quais o "SegVersionId" é diferente do "SegVersionId" armazenado na memória e que corresponde às atualizações anteriores. Se a transferência é em função da corrupção de certos segmentos no decodificador, a versão recuperada é a mesma que aquela que já está presente, e os segmentos identificados como corrompidos serão recuperados.

Uma quinta etapa consiste em verificar a integridade de cada um dos segmentos recuperados graças ao CRC de cada segmento.

Uma sexta etapa consiste em armazenar os segmentos transferidos na memória.

Embora a modalidade da invenção baseie-se na es-

trutura dos decodificadores de televisão digital, é óbvio aos versados na técnica que a invenção pode ser aplicada a todos os dispositivos que tenham capacidades de atualização de seus softwares residentes através de uma rede de distribuição de dados.

REIVINDICAÇÕES

1. Método para construir uma mídia de distribuição que contém uma pluralidade de versões do software residente destinada a uma pluralidade de dispositivos, **CARACTERIZADO** pelo fato de que compreende pelo menos as seguintes etapas:

identificar na dita pluralidade de versões de software residente, das partes comuns para pelo menos duas versões de software residente pela análise do conjunto de parâmetros (PT, PV, PR, US) definindo seu destino associado a cada parte do software residente;

dividir as diferentes versões de software residente em segmentos, chamados segmentos de software, e para cada um dos ditos segmentos de software, marcar cada segmento de software com o dito conjunto de parâmetros (PT, PV, PR, US) definindo seu destino associado a uma parte a partir da qual o dito segmento de software se origina;

para cada segmento de software de uma parte de um software residente identificado como comum, chamado segmento comum, marcar o dito segmento comum definindo em um valor específico pelo menos um parâmetro do dito conjunto de parâmetros, o valor específico que indica que o dito segmento comum é destinado a todas as plataformas de hardware;

montar o dito conjunto de parâmetros e os ditos segmentos de software dentro de uma mídia de distribuição incluindo os segmentos comuns apenas uma vez dentro da mídia de distribuição.

2. Método, de acordo com a reivindicação 1, **CARACTERIZADO** pelo fato de que o dito valor específico é

atribuído a pelo menos um parâmetro do dito conjunto de parâmetros que é representativo de um identificador de produto.

3. Método, de acordo com a reivindicação 2, **CARACTERIZADO** pelo fato de que a pluralidade de dispositivos é uma pluralidade de dispositivos de recepção de televisão digital.

4. Método, de acordo com a reivindicação 3, **CARACTERIZADO** pelo fato de que a mídia de distribuição é um fluxo contínuo de transporte MPEG.

5. Método para adquirir toda ou parte de uma versão do software residente através de um dispositivo que tem meios de acesso a uma mídia de distribuição distribuída por uma rede de distribuição, **CARACTERIZADO** pelo fato de que, a dita mídia compreende uma pluralidade de segmentos que compõe uma pluralidade de versões do software residente e sendo construído conforme definido no método da reivindicação 1, o método compreendendo pelo menos as seguintes etapas:

adquirir conjuntos de parâmetros descrevendo os segmentos incluídos na mídia de distribuição (E1);

identificar na mídia de distribuição de pelo menos um segmento de software destinado ao dito dispositivo por análise do conjunto associado de parâmetros (PT, PV, PR, US) (E3);

baixar o dito segmento identificado dentro do dispositivo a partir da dita mídia de distribuição (E4);

armazenar o dito segmento (E6) na memória.

6. Método, de acordo com a reivindicação 5,

CARACTERIZADO pelo fato de que a etapa de identificação na mídia de distribuição compreende uma etapa de comparação de um conjunto de identificadores (PT, PV, PR, US) que identificam em cada segmento os dispositivos visados pela versão do software residente a qual o segmento pertence, e um conjunto de identificadores residentes no dispositivo e codificação do seu tipo e uso.

7. Método, de acordo com a reivindicação 6, **CARACTERIZADO** pelo fato de que o conjunto de identificadores contém pelo menos um identificador de plataforma (PT, PV) que identifica o tipo específico da versão do hardware do dispositivo, um identificador de produto (PR) que identifica a operadora que opera o dispositivo e um identificador de uso (US) que identifica o uso planejado do dispositivo.

8. Método, de acordo com a reivindicação 7, **CARACTERIZADO** pelo fato de que o dispositivo é um receptor de televisão digital e a mídia de distribuição é um fluxo contínuo de transporte MPEG.

9. Mídia de distribuição de uma pluralidade de versões do software residente projetada para uma pluralidade de dispositivos, **CARACTERIZADA** pelo fato de que a dita mídia de distribuição tenha sido construída conforme definido no método da reivindicação 1.

10. Mídia, de acordo com a reivindicação 9, **CARACTERIZADA** pelo fato de que cada segmento tem um conjunto de identificadores (PT, PV, PR, US) que identifica em cada segmento os dispositivos visados pela versão do software residente à qual o segmento pertence.

11. Mídia, de acordo com a reivindicação 10, **CARACTERIZADA** pelo fato de que o conjunto de identificadores contém pelo menos um identificador de plataforma (PT, PV) que identifica o tipo específico de versão do hardware do dispositivo, um identificador de produto (PR) que identifica a operadora que opera o dispositivo e um identificador de uso (US) que identifica o uso planejado do dispositivo.

12. Mídia, de acordo com a reivindicação 11, **CARACTERIZADA** pelo fato de que tem a forma de um fluxo contínuo de transporte MPEG.

13. Dispositivo que tem meios de acesso a uma mídia de distribuição que contém uma pluralidade de versões do software residente projetadas para uma pluralidade de cinco dispositivos e sendo construído conforme definido no método da reivindicação 1, **CARACTERIZADO** pelo fato de que estas versões são divididas em uma pluralidade de segmentos na mídia de distribuição, o dito dispositivo tendo:

meios para determinar os segmentos que são parte de uma versão do software residente planejado para ele entre a pluralidade de segmentos presente na mídia de distribuição através da análise do conjunto de parâmetros associados (PT, PV, PR, US),

meios para baixar os segmentos presentes na mídia de distribuição, e

meios para armazenar esses segmentos.

14. Dispositivo, de acordo com a reivindicação 13, **CARACTERIZADO** pelo fato de que o dispositivo tem um conjunto de identificadores (PT, PV, PR, US) que codifica seu tipo e

uso e meios de comparação deste conjunto de identificadores com um conjunto de identificadores, cada um correspondendo a cada segmento presente na mídia de distribuição e que identifica os dispositivos visados pela versão de software residente à qual o segmento pertence.

15. Dispositivo, de acordo com a reivindicação 14, **CARACTERIZADO** pelo fato de que os meios de acesso a uma mídia de distribuição são meios de acesso a um fluxo contínuo de transporte MPEG.

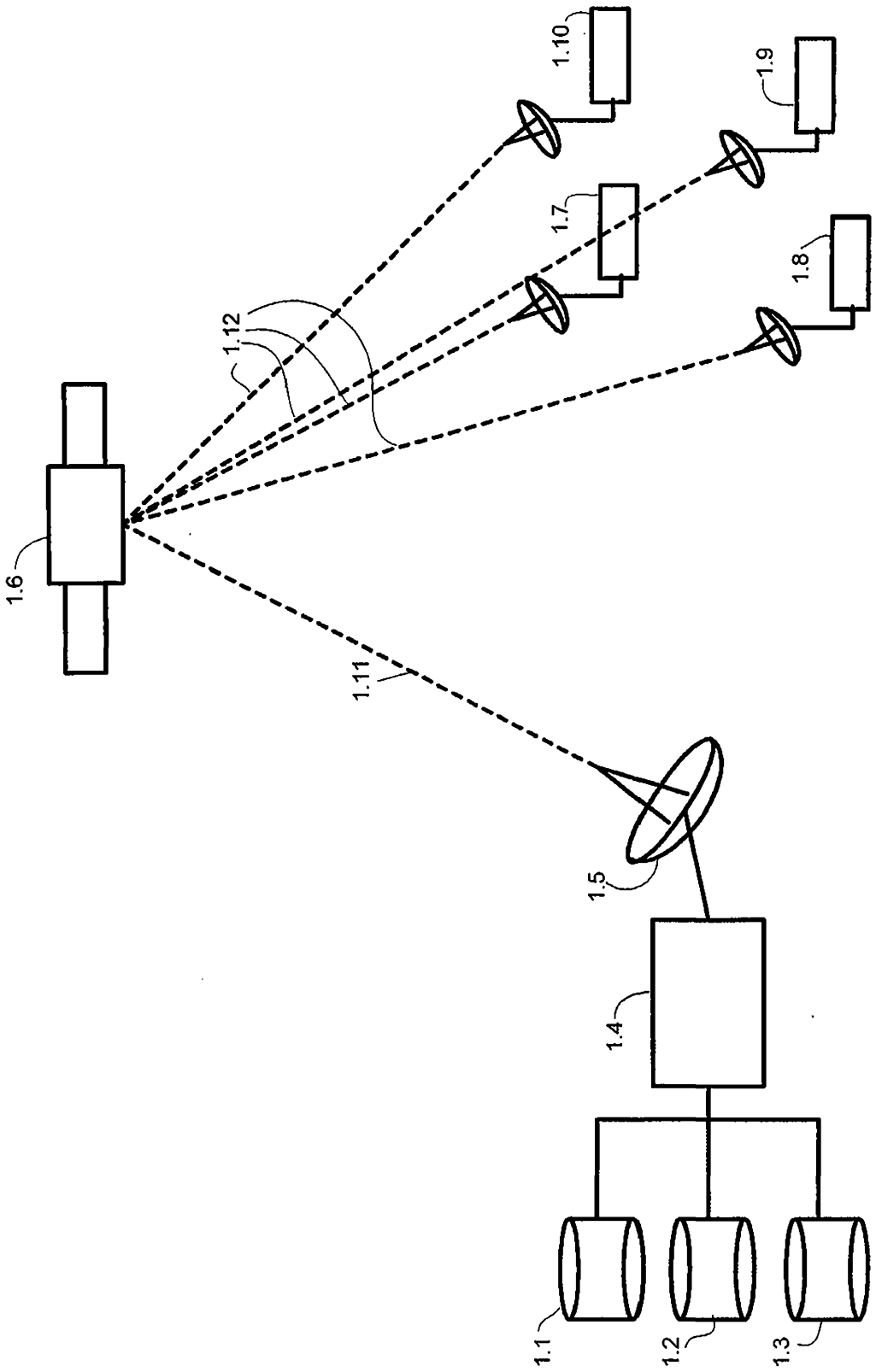

Fig. 1

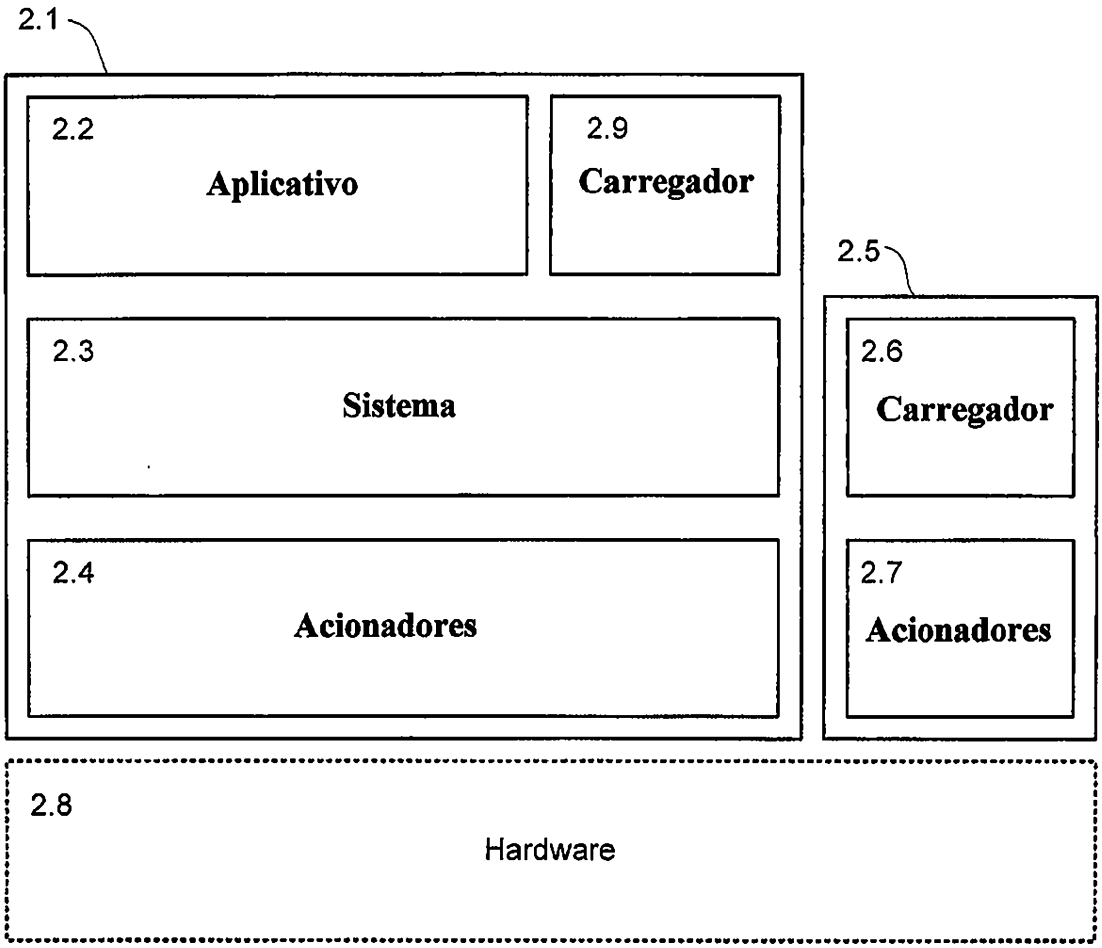
2.1
2.2
Aplicativo
2.9
Carregador
2.5
2.3
Sistema
2.6
Carregador
2.4
Acionadores
2.7
Acionadores
2.8
Hardware

Fig. 2

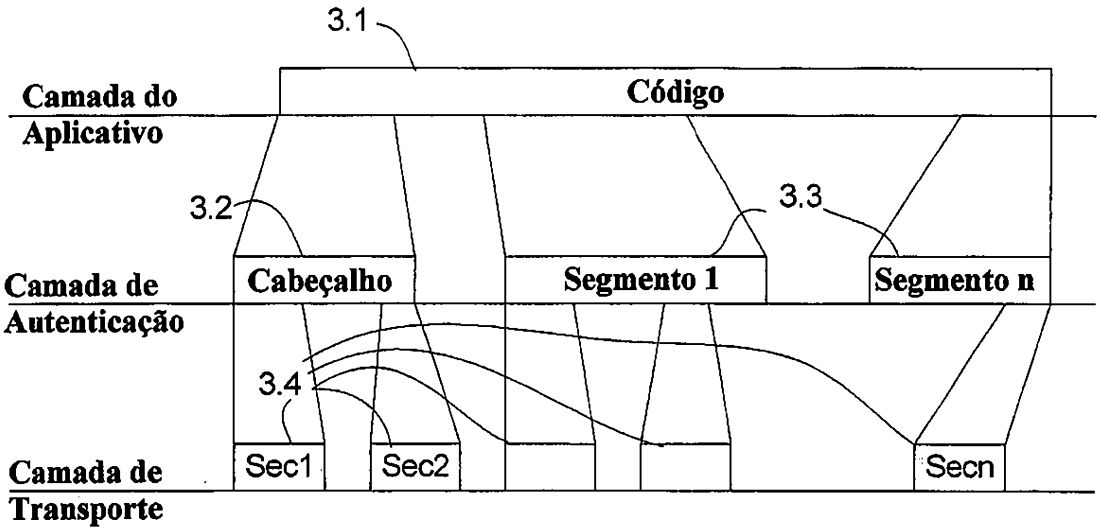
3.1
Camada do Aplicativo
Código
3.2
3.3
Camada de Autenticação
Cabeçalho
Segmento 1
Segmento n
3.4
Camada de Transporte
Sec1
Sec2
Secn

Fig. 3

Cabeçalho		
Campo	**Tamanho em bytes**	**Descrição**
MagicNumber	4	
HeaderSize	4	Tamanho do Cabeçalho
OUI	3	Identificador Exclusivo de Organização (DVB)
Não usado	1	
KeyIndex	2	Índice chave público para verificar a assinatura
FlowVersionId	2	Informação da versão do fluxo
EncryptedSeed	16	Exigido para decifrar por AES
InitialValue	16	Exigido para decifrar por AES
Não usado	4	
NumberOfSegments	4	Número de segmentos
For(s=0;s< NumberOfSegments;s++){		
SegmentNb	2	Número do segmento
PlatformType (PT)	2	Tipo de Plataforma de um ponto de vista do Hardware
PlatformVersion (PV)	2	Versão do hardware (máscara de bit)
ProductId (PR)	2	Caracteriza o produto (ou cliente)
UsageId (US)	1	Uso do segmento
SegmentVersionId	1	Número da versão do segmento
SegmentType	1	Tipo do segmento
SegmentId	1	Identificador do segmento
CrcMemoryType	1	Tipo do CRC no segmento armazenado na memória 0x0 = Sem CRC 0x2 = CRC32St20 0x3 = CRC32Mpeg2
HashMemoryType	1	Tipo de Dispersão no segmento armazenado na memória 0x0 = Sem Dispersão 0x1 = SHA
Não usado	2	
SegStorageAddress	4	Endereço de armazenamento do segmento na memória rápida
SegSizeMemory	4	Tamanho do segmento na memória (não comprimido)
CrcMemory	4	CRC do segmento armazenado na memória
HashMemory	20	Dispersão do segmento armazenado na memória (não comprimido)
SegSizeFlow	4	Tamanho do segmento no fluxo (na prática transmitido depois de possível compressão e encriptação)
CrcFlowType	1	Tipo de CRC no segmento transportado no fluxo (ver CRCMemoryType)
CompressionType	1	Tipo de compressão 0x0 = sem compressão 0x1= compressão modificada derivada do gzip
EncryptionType	1	Tipo de encriptação 0x0 = sem encriptação 0x1 = encriptação AES
Não usado	1	
CrcFlow	4	CRC do segmento transportado no fluxo
Não usado	4	
}		
Assinatura	128	Informação de autenticação calculada para todos os campos anteriores

Fig. 4

Seção MPEG		
Campo	Tamanho em Bits	Descrição
TableId	**8**	
SectionSyntaxIndicator	1	Não usado
'0'	1	Não usado
Reservado	2	
SectionLength	**12**	Ver DVB
SegmentNb	**16**	Número do segmento
Reservado	2	
VersionNumber	5	Não usado
CurrentNextIndicator	1	Não usado
SectionNumber	8	= 0x00
LastSectionNumber	8	= 0x00
OUI	**24**	Identificador Exclusivo de Organização
SectionSyntaxVersion	**8**	Versão da sintaxe da seção
FlowVersionId	**16**	Versão de fluxo da informação
Padding	16	
SegmentOffset	**32**	Indica o endereço relativo do campo de dados no fluxo
For(n=0;n< N;n++){		
Data	**N*8**	
}		
Crc32	**32**	

Fig. 5

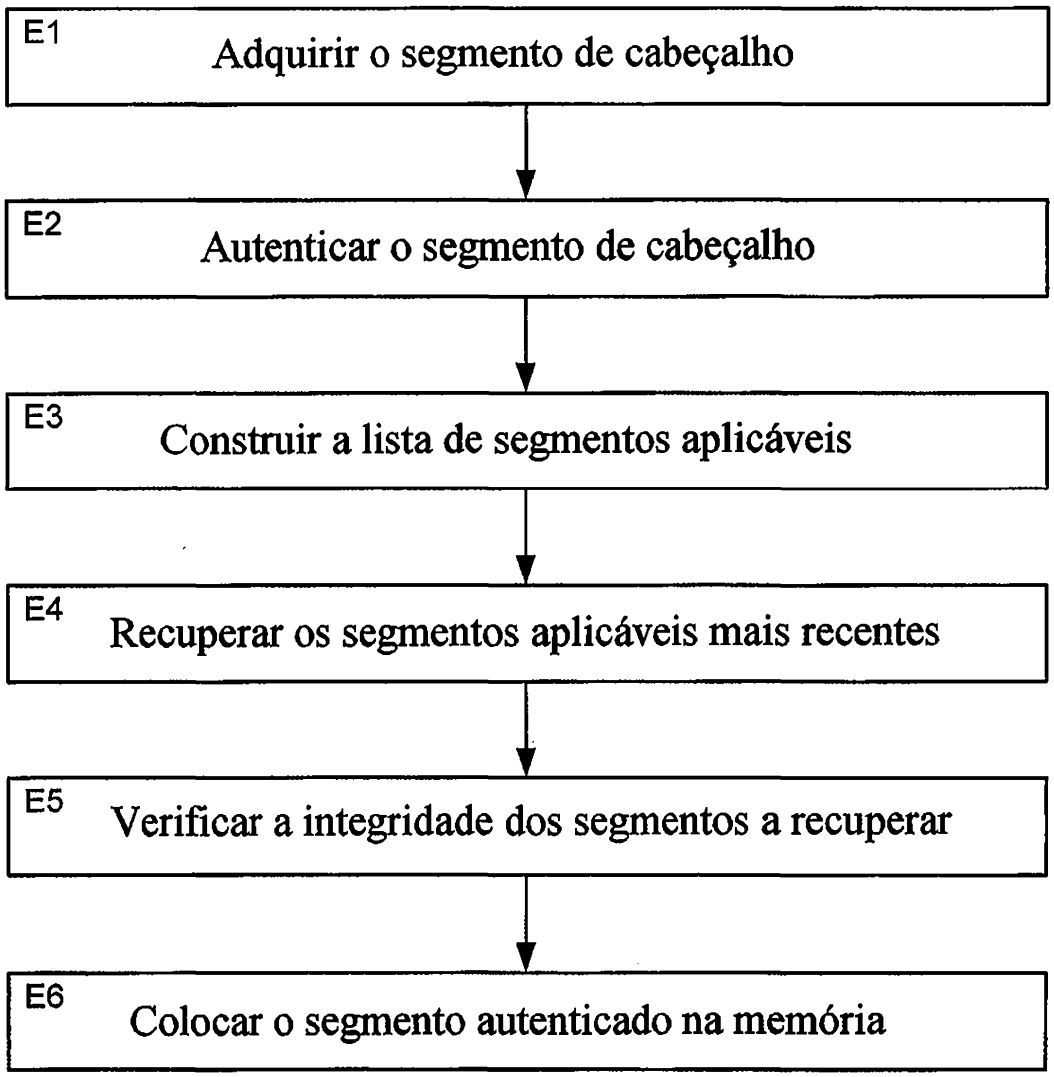

Fig. 6

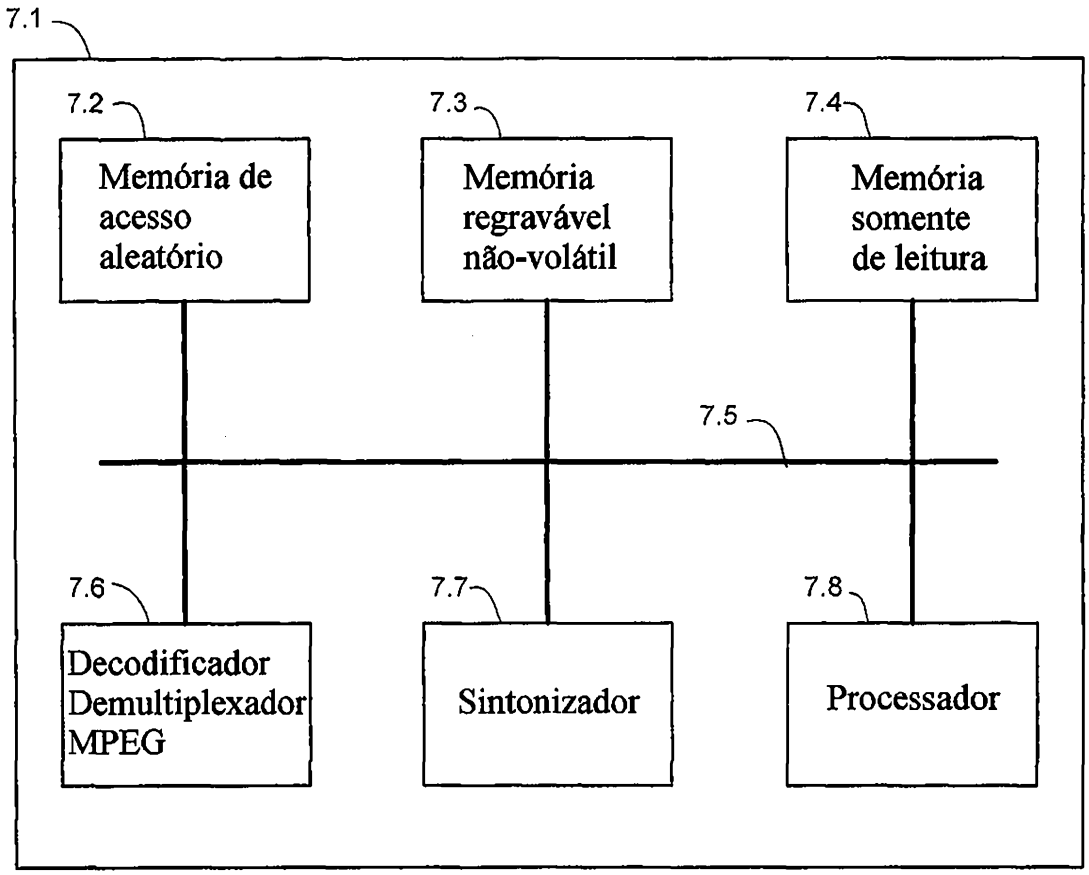

Fig. 7