# Extrator de dependências condicionais para sistemas de comunicação multímidias

*Luiz Fernando Rust da Costa Carmo e Jean-Pierre Courtiat*

NCE-UFRJ
Caixa Postal 2324, 20001-970
Rio de Janeiro-RJ
E-mail: rust@nce.ufrj.br

LAAS-CNRS
7 Avenue du Colonel Roche, 31077
Toulouse Cedex - France
E-mail: courtiat@laas.fr

**Resumo**

O projeto de aplicações multimidias envolve problemas de sincronização temporais e espaciais relacionados ao processamento, transporte, armazenamento, restituição e apresentação de diversas midias combinadas, como dados, sons, imagens e videos. Esse artigo aborda a sincronização nos sistemas distribuidos multimidias e se interessa mais especificamente aos aspectos relativos a comunicação, ou seja, de como garantir a sincronização entre diferentes midias de maneira a possibilitar uma apresentação coordenada de um objeto multimidia. Para isso definiu-se o extrator de dependências condicionais, que é baseado na identificação de relações causais, também chamadas de relações de dependências, entre as unidades de informação de um ou vários fluxos de um objeto multimidia. O objetivo maior é garantir que estas relações causais, especificadas a nivel de usuário, sejam satisfeitas no instante de restituição do objeto multimidia (entrega dos fluxos à aplicação). Neste artigo é discutido detalhadamente o extrator de dependências condicionais, e apresentado alguns exemplos simples de utilização.

**Abstract**

The design of a distributed multimedia application involves both temporal and spatial synchronization problems related to the processing, transport, storage, retrieval and presentation of data, sound, still images and video. Within this general framework, the paper aims to define a general-purpose multimedia synchronization tool for communication purposes, the so-called conditional dependency extractor, which may be applied to solve both intra- and inter-stream synchronization issues. The conditional dependency extractor is based on the identification of causal relations, also called dependency relations, among information units of one or several streams; it has the purpose of guaranteeing that these causal relations, expressed at the user's level, are satisfied when delivering the streams. This conditional dependency extractor is analyzed in depth and simple examples are provided.

## 1. Introdução

Os últimos dez anos constituiram um periodo de formidáveis progressos em matéria de tecnologia e técnicas de comunicação. Com estes suportes, a transferência de informações de som e video se tornaram possiveis nos sistemas inicialmente restritos aos dados numéricos e textuais. Mais especificamente, as aplicações ditas multimidias podem ser desenvolvidas utilizando as caracteristicas destas redes e dos novos recursos informaticos de um sistema de tratamento multimidia.

A sincronização multimidia pode ser definida como sendo a tarefa de coordenação, escalonamento e apresentação de objetos multimidias no tempo e no espaço. A finalidade de um sistema de comunicação multimidia consiste em preservar um certo numero de propriedades (ou dependências) durante a transferência

de diferentes midias entre dois ou diversos locais. O tempo global de uma transferência e constituido de um tempo de produção da informação (amostragem, compressão, filtragem, etc.), de um tempo de transporte da informação e finalmente de um tempo de restituição de uma midia. Uma das motivações da função de sincronização é a presença do *jitter*, que pode ser caracterizada pela variação estatistica do tempo de transferência global, que afeta uma midia em particular. Desta forma, verifica-se que o tempo de transferência depende tanto do sistema de comunicação, quanto dos tratamentos realizados nas extremidades pelos sistemas fontes e destinos, que por sua vez dependem das caracteristicas de tempo-real e da carga pontual dos sistemas operacionais utilizados. Essa definição de sincronização agrupa varios tipos de problemas relativos a informação propriamente dita, que induzem necessidades especificas no sistema de comunicação: (i) como as dependências temporais simples podem ser garantidas durante a transferência de uma certa midia; (ii) como as dependências temporais estruturadas podem ser garantidas de forma que as inter-relações temporais entre as diferentes midias de um objeto multimidia possam ser contempladas durante uma restituição coordenada destas midias em um dado local, ou em um conjunto de localidades distintas.

Uma informação multimidia pode ser modelada por um grupo de fluxos (denominado de feixe multimidia), sendo que um fluxo é ele próprio definido como sendo uma sequência temporal de unidades de informação. Baseado nesta definição, a problemática da sincronização multimidia é composta por duas grandes vertentes, que são respectivamente a sincronização intra-fluxo e a sincronização inter-fluxo:

- a sincronização intra-fluxo é destinada a gerência de um único fluxo (por exemplo um fluxo de audio ou vídeo) durante a sua transferência pela rede; essencialmente, ela deve garantir uma compensação do jitter introduzido pelo serviço de comunicação;
- a sincronização inter-fluxo é uma generalização da sincronização intra-fluxo e corresponde ao respeito das relações temporais entre as unidades de informação que compõem os fluxos que devem ser sincronizados (por exemplo, a sincronização entre um fluxo de audio e um fluxo de video).

Este trabalho emprega diferentes estratégias de sincronização que se adaptam a estas necessidades. Basicamente existem duas diferentes abordagens (complementares): uma que é baseada nas dependências temporais (estratégias clássicas de compensação do jitter) e outra que é baseada nas dependências condicionais. A estratégia de sincronização baseada nas dependências condicionais aborda a sincronização intra-fluxo e inter-fluxo através da exploração das relações semânticas explícitas entre as unidades de informação, de um ou vários fluxos de um feixe multimidia. Este tipo de abordagem reforça o poder de sincronização da estratégia temporal. A implementação destes serviços de sincronização se faz através dos denominados algoritmos de restritores, que por sua vez compõem os mecanismos de entrega condicional [Carmo 95].

## 2. Motivação

As propriedades de sequenciamento e confiabilidade nem sempre podem ser garantidas durante a transferência de um fluxo pelo sistema de comunicação. De acordo com o tipo de serviço garantido pelo sistema de comunicação, o conhecimento das relações de dependências entre as unidades de informação de um mesmo fluxo se transforma em um trunfo muito importante. Isto é particularmente verdadeiro quando as unidades de informação apresentam fortes relações semânticas entre elas. A titulo de ilustração, consideremos o exemplo de uma transferência de um fluxo de vídeo codificado segundo a tecnologia MPEG [ISO 93a]: neste caso as unidades de informação são os quadros do fluxo de vídeo que empregam uma técnica de codificação diferencial: em outras palavras, alguns quadros contém apenas as modificações da imagem em relação aos quadros precedentes. Neste tipo de situação é interessante efetuar uma entrega seletiva à aplicação dos quadros recebidos pelo sistema de comunicação, impedindo a entrega de um quadro codificado à partir de um ou vários quadros anteriores do mesmo fluxo que por algum motivo não puderam ser entregues à aplicação.

As dependências condicionais definidas anteriormente em um contexto de um único fluxo podem ser generalizadas de forma a estabelecer relações de dependências entre as unidades de informação que pertencem a fluxos distintos dentro de um mesmo feixe. Essa generalização permite levar em conta:

i) as *dependências condicionais simples*: - neste caso, as relações de dependências podem ser representadas por um grafo orientado (sendo que um arco de uma unidade de informação $a$ em direção à $b$ significa que a entrega de $a$ esta condicionada a entrega de $b$) que não contem nenhum circuito (i.e.,

não existe nenhuma relação de dependência acoplada); nessa configuração, uma unidade de informação pode, e deve ser entregue, desde que a unidade de informação de quem ela depende tenha sido entregue; a entrega de uma unidade de informação deve igualmente contemplar os requisitos temporais associados à transferência do fluxo.

ii) as *dependências condicionais acopladas* - neste caso, as relações de dependências podem ser representadas através de um grafo que contém no mínimo um circuito, i.e., a entrega de uma unidade de informação *a* é condicionada a entrega de uma unidade de informação *b*, que por sua vez depende da entrega da unidade de informação *a*; a existência de um circuito no grafo implica à definição de um *ponto de sincronização* entre fluxos distintos de um mesmo feixe; o conceito de ponto de sincronização permite a especificação de uma entrega simultânea de um conjunto de unidades de informação que compõe o circuito; se, por uma razão ou outra, uma unidade de informação (entre aquelas associadas ao ponto de sincronização) não pode ser entregue (por exemplo, devido à impossibilidade de respeitar um certo requisito temporal), então nenhuma das unidades de informação será entregue.

As dependências condicionais entre unidades de informação são especificadas através de expressões booleanas (chamadas de *expressões de dependências* e notadas *Dexpr*) que expressam os requisitos, em termos de dependências condicionais, a garantir durante a restituição de um fluxo ou de um feixe. Consideremos por exemplo que um fluxo S1 seja especificado usando a seguinte notação:

$$S1 = {}_{t1}n1,\ n1\ {}_{t2}n2,\ n1 \wedge (\neg n2)\ {}_{t3}n3$$

Esse tipo de especificação engloba dois objetivos principais que são respectivamente:

i) a caracterização da assinatura temporal do fluxo tal como ela pode ser especificada no instante de submissão deste fluxo; neste exemplo ela é definida pelo conjunto de instantes *{t1, t2, t3}*;

ii) a definição das dependências condicionais entre as unidades de informação que devem ser respeitadas durante a restituição do fluxo no seu destino; a especificação do fluxo S1 implica que a unidade de informação *n2*, uma vez disponível (i.e., recebida do serviço de transporte) não poderá ser entregue (a aplicação) enquanto a unidade de informação *n1* não tiver sido entregue; em outras palavras, do ponto de vista da aplicação é inútil entregar *n2*, caso a entrega de *n1* não tenha sido possível; da mesma forma, a unidade de informação *n3* só poderá ser entregue se a condição $n1 \wedge (\neg n2)$ for satisfeita, o que corresponde a entregar *n1* e a não entregar *n2*.

Quando nenhuma expressão de dependências é explicitamente associada a uma unidade de informação, esta unidade pode ser entregue assim que ela esteja disponível, sem precisar avaliar nenhuma condição.

A seguir nós ilustraremos a utilização das expressões de dependências, tanto no caso de sincronização intra-fluxo (transferência de um fluxo), como no caso da sincronização inter-fluxo (transferência de um feixe), para um serviço de transporte não (completamente) confiável.

*Sincronização intra-fluxo*

Para ilustrar as dependências condicionais intra-fluxo nós escolhemos a transferência de um fluxo de vídeo que utilize a técnica de codificação diferencial predita da norma MPEG. As relações de dependências entre as imagens (unidades de informação) MPEG do tipo I, B e P, são ilustradas pela próxima figura:

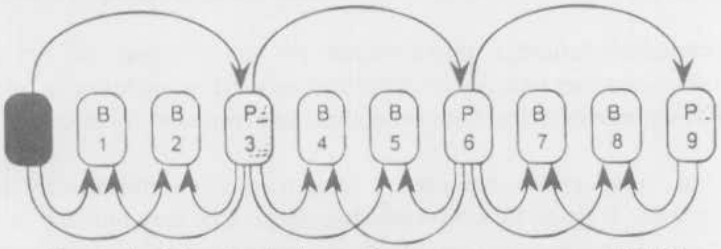


***Figura 1**- Relações de dependências entre imagens MPEG*

Esta mesma seqüência de imagens expressa de acordo com a ordem do fluxo binário, (tal como o fluxo é transferido pelo serviço de comunicação) é definida da seguinte forma:

| I | P | B | B | P | B | B | P | B | B | I | B | B | ... |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 3 | 1 | 2 | 6 | 4 | 5 | 9 | 7 | 8 | 12 | 10 | 11 | ... |

As relações de dependências condicionais intra-fluxo da figura 1 podem ser expressas através de expressões booleanas associadas aos identificadores das unidades de informação (imagens). Estas expressões de dependências caracterizam as restrições de dependências desejadas quando da restituição do fluxo de vídeo MPEG:

$$S_{MPEG} = t0\,I0,\ I0_{t1}P3,\ I0 \wedge P3_{t2}B1,\ I0 \wedge P3_{t3}B2,\ P3_{t4}P6,\ P3 \wedge P6_{t5}B4,\ P3 \wedge P6_{t6}B5,\ P6_{t}\text{-}P9,\ldots$$

Uma especificação deste tipo determina o seguinte comportamento: quando uma imagem B (usa uma codificação predita bidirecional) estiver disponível (recebida dos serviços de transporte), ela somente será entregue quando as imagens P correspondentes forem entregues; além disso, uma imagem do tipo P só é entregue se e somente se as imagens do tipo I e P correspondentes tiverem sido entregues anteriormente.

*Sincronização inter-fluxo*

As expressões de dependências condicionais inter-fluxo permitem o estabelecimento de pontos de sincronização entre os fluxos de um mesmo feixe. Consideremos o feixe da figura 2, que é composto de dois fluxos.

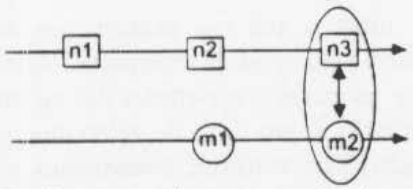


***Figura 2** - Ilustração de um ponto de sincronização*

A entrega da unidade de informação *n3* do primeiro fluxo depende da entrega da unidade de informação *m2* do segundo fluxo, e vice-versa. A presença desse ponto de sincronização impõe que as duas unidades de informação *(n3* e *m2)* sejam entregues exatamente no mesmo instante, isto é, que elas sejam mutuamente sincronizadas. No caso em que uma das unidades de informação esteja indisponível (i.e. não foi recebida pelo serviço de transporte), então nenhuma delas pode ser entregue. O feixe que captura estas propriedades de sincronização pode ser especificado da forma seguinte:

$$B = \begin{array}{l} t1\,n1,\ t3\,n2,\ m2_{t5}n3 \\ t2\,m1,\ n3_{t4}m2 \end{array}$$

Um exemplo de aplicação deste tipo de construção é dado pela transferência de um feixe composto de fluxo de slides e de um fluxo dos comentários associados a estes slides. Para cada unidade de informação do primeiro fluxo (slide) é associado uma unidade de informação do segundo fluxo (comentário correspondente), o que pode ser expresso através de expressões de dependências acopladas. Esse tipo de especificação estabelece vários pontos de sincronização:

$$B = \begin{array}{l} m1_{t1}n1,\ m2_{t3}n2,\ m3_{t5}n3,\ m4_{t}\text{-}n4,\ldots \\ n1_{t2}m1,\ n2_{t4}m2,\ n3_{t6}m3,\ n4_{t8}m4,\ldots \end{array}$$

Pode-se observar neste caso que o mais importante não é de contemplar os requisitos temporais de cada fluxo, mas sim de garantir a sincronização no instante da apresentação de cada slide e do seu respectivo comentário.

## 3. Contexto arquitetural

A integração das aplicações multimídias em um sistema de comunicação implica na definição de novos serviços adaptados as suas necessidades específicas. Essa integração deve ser realizada dentro de um contexto de trabalho que considere por um lado as novas características das aplicações, e por outro lado, as novas potencialidades dos sistemas de comunicações emergentes.

A nossa proposta arquitetural segue em linhas genéricas as atuais tendências das principais arquiteturas voltadas aos sistemas distribuídos multimídias, propondo uma distinção entre funcionalidades de transporte e funcionalidades de sincronização. Por outro lado, faz-se igualmente a distinção entre nível de sincronização e nível de orquestração. O primeiro disponibiliza funções de controle das relações causais e temporais que vão possibilitar uma restituição coordenada das informações, durante a transferências de vários fluxos, provenientes de um ou vários emissores, em direção a um receptor (ou, por difusão, a vários receptores). O nível de orquestração agrupa as funções de gerência de conexões e de associações mutipartes (coordenação de uma sessão multimídia), bem como as funções de controle dinâmico de grupo hierárquicos.

Em [Carmo 95] foram propostos novos mecanismos de sincronização (entende-se por mecanismos de sincronização, o conjunto de algoritmos, protocolos e recursos necessários a uma transferência de informações multimídias que seja capaz de preservar as suas propriedades iniciais), denominados mecanismos de entrega condicional, implementados na entidade de sincronização situada no sistema receptor. Para cada unidade de informação recebida da camada de transporte, o mecanismo determina se esta UI deve ser entregue ou não, em função da análise dinâmica da sua expressão de dependências.

O nível de sincronização repousa sobre o que nós chamaremos de subsistema de comunicação, e que é simbolizado internamente a arquitetura pelo nível de transporte. Na nossa opinião, o papel de uma camada de transporte multimídia é de aproveitar as novas propriedades das informações multimídias (por exemplo a perda de alguns quadros durante a transferência de um fluxo de vídeo não implica necessariamente em conseqüências prejudiciais a qualidade da restituição) para construir mecanismos protocolares mais eficientes que suportem tanto os requisitos de alta vazão como os requisitos de atrasos limitados. Nós podemos dizer que durante a transferência contínua de um fluxo, o transporte faz tudo que é possível em funções dos parâmetros de qualidade de serviços especificados pelo usuário e dos serviços de rede adjacentes. Por sua vez, a sincronização leva em consideração os novos requisitos dos usuários tais como a sincronização intra- e inter-fluxo, bem como as manipulações temporais.

Até agora nós mostramos o interesse da estratégia de sincronização baseada na avaliação das expressões de dependências condicionais. Porém, nada foi dito sobre como traduzir esta estratégia de sincronização sob a forma de um serviço de comunicação, em outras palavras, de como estabelecer as expressões de dependências que serão empregadas durante a transferência de um feixe multimídia.

Nos exemplos citados anteriormente, as expressões de dependências foram explicitamente especificadas a partir de um completo conhecimento das relações temporais e causais entre as unidades de informação dos feixes multimídias. No entanto, para algumas aplicações multimídias é necessário viabilizar a expressão da qualidade de sincronização de uma forma bem mais abstrata, já que as relações de dependências temporais e causais entre as unidades de informação podem não ser conhecidas a priori.

Para isso definiu-se o extrator de dependências condicionais, que pode ser visualizado como uma ferramenta de auxílio a especificação da qualidade de serviço de sincronização baseado na avaliação de expressões de dependências condicionais. O principal objetivo do extrator de dependências condicionais é permitir um mapeamento automático de cenários de apresentações multimídias em esquemas de dependências condicionais que vão controlar a transferência das unidades de informação dos fluxos envolvidos.

Uma outra funcionalidade provida pelo extrator consiste na viabilização do desenvolvimento de uma biblioteca de esquemas de relações de dependências genéricos (por exemplo: esquemas de relações de dependências para fluxos codificados em MPEG). A partir destes serviços, o usuário poderá simplesmente indicar aos serviços de sincronização, o tipo de codificação dos fluxos, e a partir desta informação as relações de dependências são deduzidas automaticamente.

Dessa forma, é proposta uma técnica de controle da qualidade de serviço de sincronização, que permite a uma aplicação multimídia escolher em qual nível de abstração ela deseja se enquadrar, tal como uma usuária dos serviços de sincronização. Esse abordagem procura satisfazer quatro objetivos principais, que são respectivamente:

i) oferecer a aplicação uma forma mais abstrata de especificar a qualidade de serviço de sincronização esperada na transferência de um feixe multimídia;
ii) disponibilizar um conjunto de facilidades para passar automaticamente de um cenário de apresentação multimídia (ex: especificado através de MHEG) para um esquema de dependências condicionais que controla as transferências das unidades de informação;
iii) fornecer uma biblioteca de esquemas de relações de dependências genéricas (ex: esquemas de relações de dependências para fluxos de vídeo codificados em MPEG), de tal forma que o usuário possa simplesmente indicar ao serviço de sincronização o tipo de codificação dos fluxos e, a partir dele, deduzir as relações de dependências;
iv) estabelecer equivalências funcionais entre os serviços de entrega condicional e outros serviços de sincronização baseados em outras técnicas de sincronização clássicas, tais como os marcadores e os canais de sincronização [STE 90]; a idéia de base é disponibilizar a mesma qualidade de serviço fornecida pelas outras técnicas de sincronização porém utilizando a estratégia de sincronização por entrega condicional.

O extrator de dependências condicionais é constituído de dois elementos distintos, representados por duas unidades funcionais: (i) *a unidade de especificação das expressões de dependências* fornece as funcionalidades que permitem a uma aplicação gerenciar ela mesma as expressões de dependências condicionais associadas a um fluxo ou feixe; e (ii) *a unidade de geração das expressões de dependências* permite que a aplicação especifique a sincronização de uma maneira bem mais abstrata, de forma que a geração das expressões de dependências seja inteiramente efetuada pelo provedor do serviço de sincronização.

## 4. Unidade de especificação das expressões de dependências

Esta unidade possibilita três diferentes abordagens para o controle da especificação das expressões de dependências: controle através de expressões de dependências unitárias, controle por esquema de dependências condicionais e controle por macro-expressões de dependências. O uso de expressões de dependências unitárias corresponde ao nível de abstração mais baixo, e o uso de macro-expressões de dependências representa o mais alto nível de abstração.

### 4.1 Controle por expressões de dependências unitárias

A aplicação multimídia deve fornecer dinamicamente todas as expressões de dependências associadas às unidades de informação do feixe em questão. Para cada submissão de uma nova UI, a aplicação fornece a expressão de dependências correspondente. A vantagem desta abordagem consiste de um domínio completo, por parte da aplicação, da degradação residual permitida quando da perda de UI's. A partir do momento em que seja aumentado sensivelmente o nível de abstração da especificação das dependências condicionais, ocorre uma degradação do controle dos efeitos causados numa transferência com perdas.

### 4.2 Controle por esquemas de dependências

Esse tipo de controle possibilita ao usuário especificar antecipadamente o conjunto de expressões de dependências que será utilizado durante a transferência do fluxo ou feixe. Consequentemente, a transferência de um feixe é composta de duas fases distintas: a fase de definição do esquema de dependências e em seguida a transferência de dados propriamente dita, durante a qual cada evento de submissão de uma UI provoca uma consulta ao esquemas de dependências especificado antecipadamente. Esta consulta deve gerar (ou não) a expressão de dependência correspondente, que acompanhará a transferência da UI até a entidade de sincronização receptora.

Este tipo de abordagem de especificação facilita a utilização das dependências condicionais por parte das aplicações do tipo contínua. Consideremos, como exemplo, a transferência de um fluxo de vídeo codificado em

MPEG-I de uma aplicação de teleconferência. Neste caso a aplicação conhece as relações de dependências que devem ser associadas a transferência. Através do emprego desta técnica de especificação, a aplicação poderá especificar todos os requisitos antecipadamente, via um esquema de dependências, e em seguida submeter os quadros gerados diretamente pelo codificador aos serviços de sincronização, sem ter que adicionar as expressões de dependências.

Uma outra vantagem desta técnica é permitir a criação de uma biblioteca de esquemas de dependências. Dessa forma, uma aplicação poderá tanto recuperar esquemas de dependências pré-concebidos, como guardar outros esquemas que apresentem um interesse genérico.

**4.2.1 Composição de um esquema de dependências** - Um esquema de dependências condicionais é inteiramente concebido sobre a definição de variável unitária (*vu*). Uma variável unitária especifica um lugar vazio que será preenchido por uma unidade de informação durante a fase de transferência de dados de um feixe. Existem duas diferentes maneiras de especificar uma variável unitária: referência temporal e referência de ordem. Uma *vu* pode representar tanto um argumento de uma expressão de dependências quanto uma unidade de informação condicionada por uma expressão de dependências.

Um esquema de dependências corresponde ao conjunto de expressões de dependências condicionais (utilizando apenas *vu*'s) pré-estabelecidas para a transferência de um feixe. A figura 3 descreve um exemplo de utilização de um esquema de dependências durante a transferência de um fluxo. A figura 3(i) ilustra o esquema de dependências que é composto de um conjunto de *Dexpr*'s construídas sobre lugares vazios (variáveis unitárias *a,b,c,d,e,f*).

A figura 3(ii) descreve o fluxo submetido ao serviço de sincronização, e a figura 3(iii) mostra o procedimento dinâmico de preenchimento dos lugares vazios por UI's, conduzindo ao fluxo da figura 3(iv).

A especificação de um esquema de dependências condicionais é estruturada em três níveis: variável unitária, motivo e corpo.

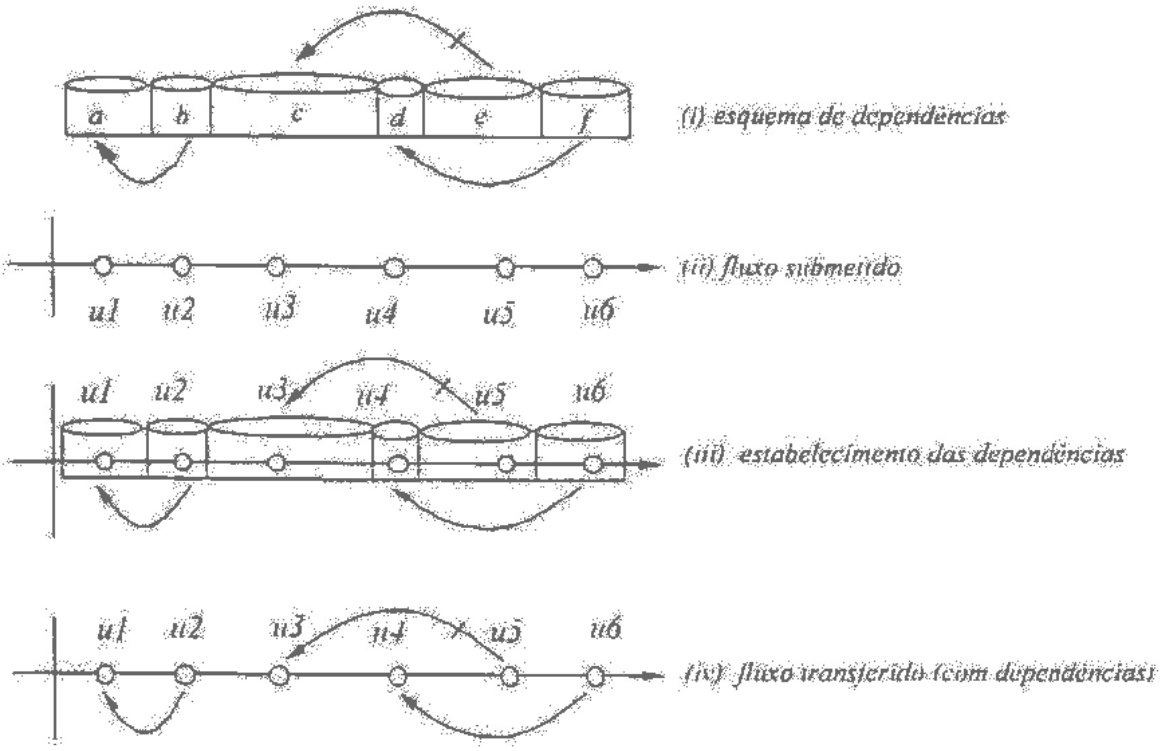


*Figura 3 - Exemplo de um esquema de dependências*

**4.2.2 Variável unitária (vu)** - Essencialmente, uma variável unitária é definida por uma referência temporal ou uma referência de ordem em relação ao início de um *motivo* (definido no próximo item) no qual ela foi declarada internamente. No caso de utilização de uma referência temporal, a aplicação deve especificar o

intervalo de tempo presumido para ocorrência da emissão da UI em questão. Caso contrario, a aplicação deve especificar o posicionamento da UI em relação ao inicio do motivo. Se a submissão da UI não ocorrer durante o intervalo de tempo associado a uma variável unitária, então o tratamento da expressão de dependência a considera como perdida. No caso de múltiplas submissões de UI's durante um certo intervalo, a *vu* assume o valor da primeira submissão.

Além destes dois tipos de referências, é previsto uma classe especial de *vu's* ligadas à gerência de eventos dos serviços de sincronização. Essa classe de *vu* se distingue das demais devido ao tratamento diferenciado que ela provoca durante a etapa de recomposição das expressões de dependências. Na verdade, estas vu's não correspondem às unidades de informação, mas a eventos que são gerenciados pelo receptor através da emissão/recepção de primitivas de serviço adequadas.

Por exemplo, uma aplicação pode condicionar a entrega de uma UI *a* à ocorrência de um evento *e*, associando a esta uma expressão de dependências que utilize como argumento uma *vu* de referência evento. Como consequência, a unidade de informação somente será entregue quando o receptor tive recebido uma primitiva de serviço que corresponda à ocorrência do evento *e*. Se uma variável unitária de referência evento está condicionada por uma expressão de dependências, então o receptor deve emitir uma primitiva de serviço à aplicação, no instante em que esta expressão for resolvida.

As variáveis unitárias de referência evento oferecem uma maneira simples e elegante de uma aplicação multimídia levar em conta eventos externos durante a transferência de um feixe. Um exemplo de uma aplicação em potencial desta funcionalidade é a manipulação de um objeto multimídia que contenha um menu, representado pela norma MHEG [ISO 93b] através do objeto *selection*. Este objeto disponibiliza ao usuário, uma funcionalidade de definição de um menu e sua posterior associação (através de objetos *link*) à apresentação de determinadas informações (*presentables*).

A definição completa de uma variável unitária necessita além das informações de referência (*reference*) de dois outros atributos: *vu_id* e *stream_id*, que vão, respectivamente, fornecer um identificador para a variável em questão e especificar o fluxo ao qual ela pertence. Em seguida é apresentado a especificação detalhada de uma variável unitária:

```
variável unitária (vu)
Attribute:      vu_id
Attribute:      reference
        Attribute:      ref_type
        Constraint:     ref_type = ordre
                Attribute:      position
        Constraint:     ref_type = temporal
                Attribute:      time-interval
        Constraint:     ref_type = event
                Attribute:      event_id
Attribute:      stream_id
```

**4.2.3 Motivo** - Um motivo representa um comportamento especifico de uma parte do corpo de um esquema de dependência, sendo composto de uma lista de variáveis unitárias, de um conjunto de expressões de dependências intra-motivo e de uma duração. Um motivo pode ser instanciado várias vezes durante a transferência de um feixe. Todas as variáveis unitárias definidas dentro de um motivo fazem referência ao começo do motivo e não ao começo da transferência de um fluxo; elas devem seguir um tipo único de referência (ordem ou temporal).

Uma expressão de dependências intra-motivo (*intra_motif_Dexpr*) efetua a ligação entre as UI's de um mesmo motivo. Tanto os argumentos das expressões de dependências, como as UI's condicionadas, são definidas através dos identificadores de variável unitária (*vu_id*). Uma expressão de dependências inter-motivo (*inter_motif_Dexpr*) destina-se à ligação entre as UI's que pertençam a motivos (ou instâncias de motivos) diferentes.

A UI condicionada deve pertencer ao motivo onde a expressão é declarada, sendo referenciada através de um identificador de variável unitária (*vu_id*). Os argumentos são referenciados através de uma tripla (*motif_id, position, vu_id*), tal que *motif_id* define o motivo em questão e *position* define a qual instância deste motivo, a variável unitária faz referência. O atributo *position* é definido por uma referência de ordem decrescente em

relação as instâncias de motivos precedentes, por exemplo *(a, 2, b)* indica a variavel *b* da penultima instância do motivo *a*.

A duração de um motivo pode ser igualmente expressa por unidades de tempo ou por um número limitado de UIs. Para manter a coerência entre as variáveis unitárias e os motivos, nos propomos que a referência escolhida para definição das variáveis unitárias do motivo seja a mesma empregada para especificar a duração de um motivo.

```
motivo (m)
        Attribute:      motif_id
        Attribute:      duration
        Attribute:      set of vu
                Attribute:      vu
        Attribute:      set of intra_motif_Dexpr
                Attribute:      conditionned_vu
                Attribute:      set of elementary_conjunctions
                        Attribute:      set of postive_premisses
                                Attribute:      vu_id
                        Attribute:      set of negative_premisses
                                Attribute:      vu_id
        Attribute:      set of inter_motif_Dexpr
                Attribute:      conditonned_vu
                Attribute:      set of elementary_conjunctions
                        Attribute:      set of postive_premisses
                                Attribute:      motif_id
                                Attribute:      position
                                Attribute:      vu_id
                        Attribute:      set of negative_premisses
                                Attribute:      motif_id
                                Attribute:      position
                                Attribute:      vu_id
```

**4.2.4 Corpo** - O corpo de um esquema de dependências define globalmente todas as expressões de dependências que são utilizadas durante a fase de transferência de um fluxo ou feixe. Um corpo é composto por uma sequência de motivos (*motif_sequence*).

*Definição* : uma sequência de motivos é definida por $Mexpr = \{(m,n) : m \in M \text{ et } n \in \aleph\}$ tal que $M$ representa o conjunto de todos os motivos e $n$ representa o número de ocorrências consecutivas de um motivo $m$. Consideremos uma expressão de motivo de um feixe $F$ especificada da seguinte forma:

$Mexpr(F) = a, b, c^3$ onde $a,b,c \in M$

Essa notação indica que a transferência do feixe F é estruturada em três motivos diferentes: inicialmente a transferência assume o comportamento expresso pelo motivo *a*, em seguida ela se comporta como o motivo *b* e finalmente ela se comporta como o motivo *c*, três vezes consecutivas.

```
corpo (scheme_body)
        Attribute:      body_id
                Attribute:      motif_sequence
                        Attribute list of
                                Attribute: motif_id
                                Attribute:      ocurrences_number
                Attribute:      set of motif
```

**Exemplo** - Para ilustrar o emprego de um esquema de dependências é utilizado mais uma vez o exemplo de transferência de um fluxo codificado em MPEG. O grafo de dependências da figura 4 ilustra os requisitos de dependências condicionais da transferência de um fluxo de video MPEG. Existem diversas maneiras de representar este grafo através de uma sequência de motivos. A maneira mais simples consiste em especificar apenas um motivo composto de doze unidades de informação consecutivas tal como definido na figura 4. Essa escolha simplifica a especificação do esquema de dependências, já que só é necessário a definição de duas expressões de dependências inter-motivos.

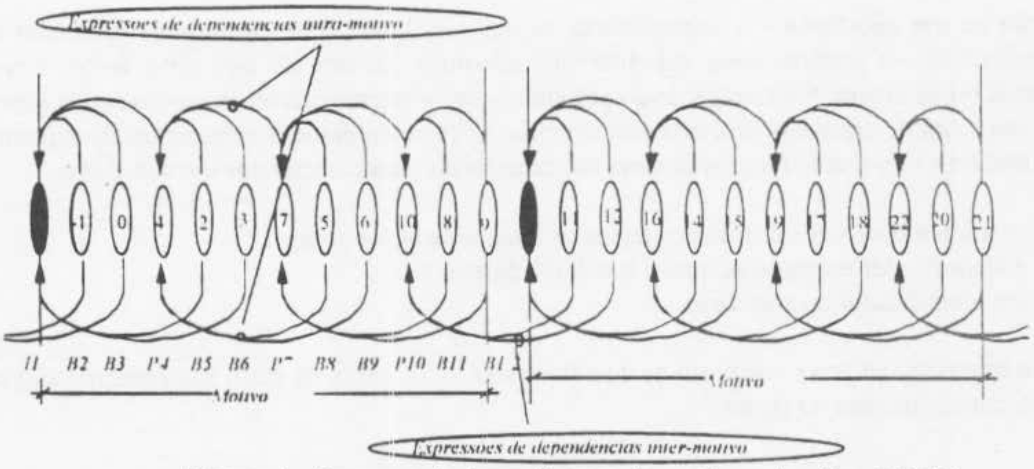


***Figura 4** - Esquema de dependências de um fluxo de vídeo MPEG*

i) o primeiro passo consiste em especificar o conjunto de variáveis unitárias que compõem o motivo:

```
vu_ens =
{
        (vu_id = I1.reference = (ref_type = ordre, position = 1), stream_id = , video),
        (B2, (ordre, 2), video), (B3, (ordre, 3), video), (P4, (ordre, 4), video),
        (B5, (ordre, 5), video), (B6, (ordre, 6), video), (P7, (ordre, 7), video),
        (B8, (ordre, 8), video), (B9, (ordre, 9), video), (P10, (ordre, 10), video),
        (B11, (ordre, 11), video), (B12, (ordre, 12), video)
}
```

ii) em seguida, acrescenta-se o conjunto de expressões de dependências intra-motivo:

```
MPEG_intra_dexpr =
{       (conditionned_vu = B2, set of elementary conjunctions = {({I1},{})}),
        (B3, {({I1},{})}), (P4, {({I1},{})}), (B5, {({I1,P4},{})}), (B6, {({I1,P4},{})}),
        (P7, {({P4},{})}), (B8, {({P4,P7},{})}), (B9, {({P4,P7},{})}), (P10, {({P7},{})}),
        (B11, {({P7,P10},{})}), (B12, {({P7,P10},{})})
}
```

iii) acrescenta-se o conjunto de expressões inter-motivo:

```
MPEG_inter_dexpr =
{       (cond_vu = B2, set of elementary conjunctions = {({(MPEG_motif.1.P10)},{})}),
        (B3, {({(MPEG_motif.1.P10)},{})})
}
```

iv) finalmente é composto o motivo e o corpo do esquema:

```
        (motif_id = MPEG_motif, duration= 12 IUs,
                set of intra_motif_dexpr = MPEG_intra_dexpr,
                set of inter_motif_dexpr = MPEG_inter_dexpr,)
        (body_id = MPEG_body, motif_sequence = {(MPEG_motif,∞)})
```

### 4.3 Controle por macro-expressões de dependências

As macro-expressões de dependências condicionais expressam os elos semânticos entre sub-fluxos de uma apresentação multimídia, i.e., entre sequências ininterruptas de UI's de um mesmo fluxo. A partir de um perfil de macro-expressões de dependências especificado antecipadamente, a unidade de especificação de expressões de dependências gera automaticamente um esquema de expressões de dependências condicionais, que será em seguida traduzido em expressões de dependências unitárias.

O interesse desta abordagem é restringir o universo de possibilidades de especificação de relações de dependências condicionais, e consequentemente fornecer uma maneira simples e bastante útil para que uma aplicação gerencie seus requisitos de sincronização. Esse nível de abstração se adapta bem a todas as aplicações que apresentam uma especificação mais orientada ao conceito de fluxo (tal como MHEG e os modelos baseados em *timeline* [HFK 95]).

**4.3.1 Especificação de um sub-fluxo** - A especificação de um sub-fluxo segue os mesmos princípios de um esquema de dependências. O usuário pode delimitar um sub-fluxo, dentro de um certo feixe, através de referências temporais ou de ordem. No caso de uma opção pelo uso de referências temporais, deve-se especificar os instantes de início e fim de um sub-fluxo; no caso contrário, deve-se especificar as posições da primeira e da última UI que delimitam o sub-fluxo. Três parâmetros são necessários para caracterizar um sub-fluxo:

(a) as fronteiras de um sub-fluxo (referências temporais ou de ordem),
(b) o identificador do fluxo ao qual o sub-fluxo pertence, e
(c) um identificador do sub-fluxo.

Consideremos o exemplo de um feixe composto de dois fluxos *S1* et *S2*, sobre os quais são especificados os dois sub-fluxos (*a* et *b*), como ilustrado na figura 5:

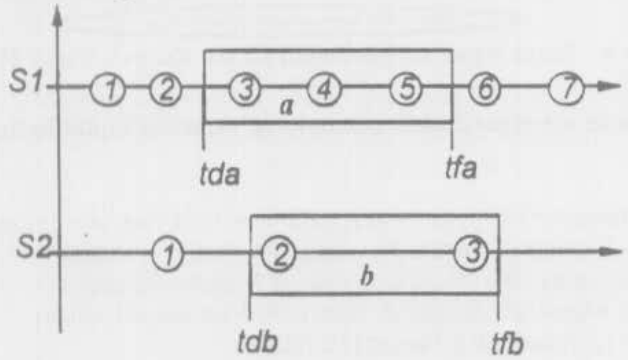


***Figura 5** - Exemplo de sub-fluxo*

Os sub-fluxos *a* e *b* podem ser especificados de duas maneiras diferentes:
(i) por referência temporal: *(S1, [tda,tfa], a)* e *(S2, [tdb,tfb], b)*, ou
(ii) por referência de ordem: *(S1, [3,5], a)* e *(S2, [2,3],b)*.

**4.3.2 Macro-expressões de dependências (*macro_dexpr*)** - Uma vez que o usuário distinguiu um conjunto de sub-fluxos dentro de um feixe, ele pode criar macro-expressões de dependências entre estes sub-fluxos. A diferença entre uma expressão de dependências e uma macro-expressão de dependências é que a última condiciona sempre o início, ou fim, de entrega de um sub-fluxo aos eventos de início, ou fim, de entrega de outros sub-fluxos (*attribute: sync_origine (begin, end)*). Consideremos alguns exemplos de expressões de dependências baseados nos sub-fluxos *a* e *b* especificados anteriormente: se a aplicação deseja que os dois sub-fluxos sejam entregues simultaneamente, basta especificar um ponto de sincronização entre o início da entrega do sub-fluxo *a* e o início de entrega do sub-fluxo *b*; se a aplicação deseja encadear o início da entrega do sub-fluxo *b* (respectivamente a) com o fim da entrega do sub-fluxo *a* (resp. b), ela deve especificar as seguintes macro-expressões de dependências (os prefixos *deb* et *fin* são utilizados para diferenciar os eventos de início e fim de um sub-fluxo, respectivamente):

$$MD\,expr^{b^{deb}} = a^{fin} \text{ e } MD\,expr^{a^{deb}} = b^{fin}$$

Estendendo a notação proposta no início deste artigo para a especificação de um feixe, de tal forma que os instantes temporais associados às UI's sejam substituídos pelas fronteiras que delimitam os sub-fluxos, pode-se representar as três situações acima por:

$$F = \begin{cases} \ldots, {}^{b^{deb}}[td_a, tf_a]^{a^{deb}}, \ldots \\ \ldots, {}^{a^{deb}}[td_b, tf_b]^{b^{deb}}, \ldots \end{cases}$$

As regras de tradução de macro-expressões de dependências em expressões de dependências unitárias são bem simples e consistem em substituir os eventos de início e fim de um sub-fluxo pelos identificadores da primeira e da última UI do sub-fluxo, respectivamente. Aplicando este procedimento aos exemplos precedentes obtêm-se as relações de dependências ilustradas na figura 6.

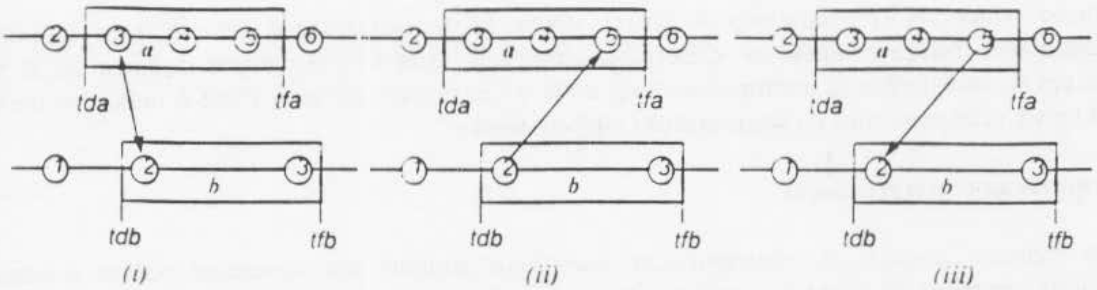


*Figura 6 - Tradução das macro-expressões de dependências*

Acrescenta-se ainda um último parâmetro (*substream_sync*) para cobrir os diferentes requisitos das aplicações no que se refere as relações de dependências entre as UI's de um mesmo sub-fluxo. Três diferentes opções são propostas: (i) frouxo (*loose*) - nenhuma relação de dependências entre as UI's, (ii) acoplado fracamente (*loosely coupled*) - todas as UI's do sub-fluxo dependem da entrega da primeira UI, et (iii) acoplado (*coupled*) - as relações de dependências são encadeadas da primeira à ultima UI. A figura 7 ilustra as três possibilidades de relações de dependências dentro de um sub-fluxo.

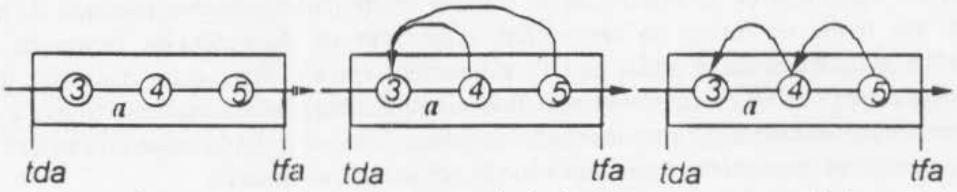


*Figura 7 - Relações de dependências dentro de um sub-fluxo*

```
macro_perfil (macro_profile)
        Attribute:      body_id
        Attribute:      set of substream
                Attribute:      substream_id
                Attribute:      substream_sync (loose, loosely_coupled, coupled)
                Attribute:      reference
                        Attribute:      stream_id
                        Attribute:      ref_type
                        Constraint:     ref_type = ordre
                                Attribute:      [first_iu, last_iu]
                        Constraint:     ref_type = temporal
                                Attribute:      time_interval
        Attribute:      set of macro_Dexpr
                        Attribute:      conditonned_sf
                                Attribute:      substream_id
                                Attribute:      sync_origine (begin, end)
                        Attribute:      set of elementary_conjunctions
                                Attribute:      set of postive_premisses
                                        Attribute:      substream_id
                                        Attribute:      sync_origine (begin, end)
                                Attribute:      set of negative_premisses
                                        Attribute:      substream_id
                                        Attribute:      sync_origine (begin, end)
```

## 5. Unidade de geração des expressões de dependências

O objetivo da unidade de geração de expressões de dependências é fornecer aos usuários do serviço de sincronização uma maneira bem mais abstrata de gerenciar os requisitos de sincronização. Essa técnica libera as aplicações multimídia da tarefa de gerência das relações de dependências, sem que elas sejam privadas de todas as funcionalidades fornecidas pela técnica de sincronização baseada nas dependências condicionais. Uma aplicação multimídia pode expressar a qualidade de serviço desejada, sem se ocupar da especificação das expressões de dependências unitárias, nem de esquemas de dependências ou de perfis de macro-expressões de dependência.

Três diferentes estratégias de sincronização inter-fluxo são fornecidas às aplicações multimídia: por marcadores, dirigida e acoplada. Estas estrategias utilizam tanto as funcionalidades providas pelos mecanismos de

sincronização temporais (ex: controle de jitter), como as funcionalidades geradas pelo mecanismo de sincronização por entrega condicional. Observa-se ainda que estas três estratégias destinam-se às seguintes configurações de transferências multimídias: uma fonte e um destino ou uma fonte e múltiplos destinos por difusão seletiva (sem requisitos de sincronização inter-destinos).

### 5.1 Sincronização por marcadores

Uma das técnicas clássicas de sincronização inter-fluxo consiste em introduzir vários marcadores de sincronização, em todos os fluxos de um feixe, durante a submissão. Esses marcadores devem ser inseridos de uma forma simultânea (inter-fluxo) e periódica. Se os fluxos são transferidos por conexões de transporte diferentes, apresentando atrasos diferentes, então os marcadores podem chegar no destino temporalmente desalinhados. Desta forma, a entidade de sincronização do receptor deverá atrasar a entrega das UI's, até o momento em que todos os marcadores submetidos simultaneamente já tenham chegado.

A implementação da estratégia de sincronização por marcadores, através do mecanismo de dependências condicionais, consiste em substituir os marcadores por UI's de controle interligadas por pontos de sincronização. Em outras palavras, a entidade de sincronização do emissor insere periodicamente (período de marcação) uma UI de controle em todos os fluxos do feixe com expressões de dependências formando um ponto de sincronização entre elas. Além disso, todas as UI's submetidas em um fluxo entre duas UI's de controle são condicionadas pela entrega da UI de controle imediatamente precedente (inter-marcação frouxa - *loose*), ou com relações de dependências encadeadas (inter-marcação acoplada - *coupled*). Enfatiza-se que as UI's de controle só servem para representar os marcadores e portanto não são repassadas ao usuário.

As próximas figuras ilustram o procedimento de geração da marcação durante uma transferência de um feixe. Os discos pretos representam as UI's de controle e os círculos brancos representam as UIs do fluxo transferido. A figura 8 ilustra as relações de dependências no caso de uma inter-marcação acoplada e a figura 9 no caso de uma inter-marcação frouxa.

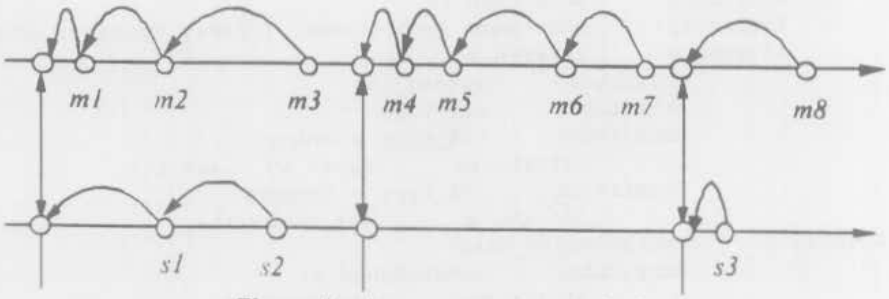


***Figura 8** - Inter-marcação acoplada*

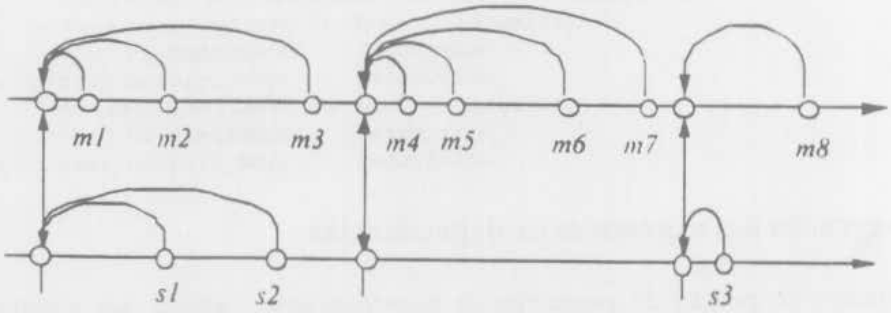


***Figure 9** - Inter-marcação frouxa*

```
marcador_perfil (marker_profile)
        Attribute:      marker_profile
                        Attribute:      period
                        Attribute:      inter-marker (loose, coupled)
```

Mesmo que o grau de sincronização gerado por essa técnica seja bastante aproximativo, existem aplicações multimídias (aplicações com requisitos temporais não muito rigorosos), às quais esta técnica é adequada.

### 5.2 Sincronização dirigida

A segunda estratégia de sincronização inspira-se nas idéias básicas da técnica do canal de sincronização [SaSh 89]. esta técnica prevê o uso de uma conexão especializada, além das conexões necessária à transferência do feixe, denominada de canal de sincronização, que deve veicular o tráfego de informações destinado ao controle da sincronização inter-fluxo. Essas informações são compostas de um parâmetro de controle da apresentação (série, paralela...) e de referências às UI's do feixe. Dessa forma o receptor deve gerenciar a restituição das UI's respeitando a ordem estabelecida pelos parâmetros de controle, sabendo que estes são deduzidos das informações de sincronização especificadas explicitamente pela aplicação, quando da submissão do feixe.

Para nós, o essencial desta técnica do canal de sincronização é a existência de um fluxo de referência responsável pelo controle da entrega das UI's do feixe. A técnica de sincronização dirigida segue esta mesma idéia através do conceito de *fluxo guia*. Nós propomos que o fluxo de informações de sincronização seja efetivamente representado por um certo fluxo do feixe (*fluxo guia*), escolhido pela aplicação (de preferência um fluxo periódico e/ou de natureza isócrona e/ou que exprime a maior densidade de UI's e o menor jitter residual permitido). Uma vez estabelecido o fluxo guia, a entrega de todas as outras UI's são condicionadas às UI's do fluxo guia. Esta técnica de geração das expressões de dependências caracteriza-se pelo seguinte comportamento:

- para cada UI de um feixe, que não pertença ao fluxo guia, é criada uma expressão de dependências que a condiciona à entrega da UI do fluxo guia submetida antes dela.

**Definição 1:** A transferência de um feixe $B$ pela estratégia de sincronização dirigida deve satisfazer a seguinte condição:

$$\forall S \in (B \setminus \{S^m\}) \; e \; \forall i \in S,\; Dexpr^i = a \text{ tal que } (a \in S^m) \; e \; \neg(\exists b \in S^m : ts^a < ts^b \leq ts^i)$$

$$\text{onde} \begin{cases} S^m \text{ é o fluxo guia do feixe } B \\ ts^a, ts^b \text{ e } ts^i \text{ são os instantes de submissão das UI's } a, b, \text{ e } i, \text{ respectivamente} \end{cases}$$

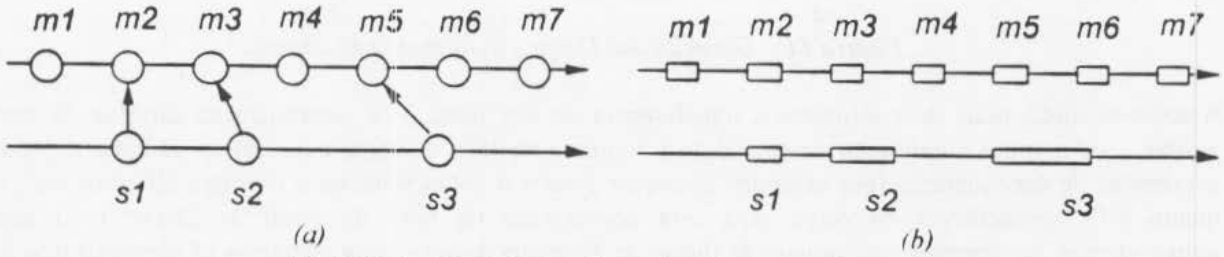


***Figura 10** - Geração das Dexpr's e diagrama de entrega*

A figura 10 ilustra o feixe entregue ao usuário de acordo com a estratégia da sincronização dirigida: os pequenos retângulos representam os jitters residuais máximos associados às UI's. O jitter máximo do primeiro fluxo (fluxo guia) é especificado diretamente pelo usuário, enquanto que o jitter do segundo fluxo é uma consequência do jitter e do período, do fluxo guia. A seguir é proposta uma maneira de calcular o jitter residual máximo de um feixe a partir do conhecimento do jitter residual e do período do fluxo guia.

**Proposição 1**: Considere uma transferência sem erros de um feixe $B$ seguindo a estratégia de sincronização dirigida, tal que $S^m$ seja escolhido como fluxo guia. Sejam $p$ e $j$ o período (ou espaçamento máximo do fluxo $S^m$) e o jitter residual do fluxo guia, respectivamente, então o jitter residual máximo associado à todos os fluxos do feixe $B$ é definido por:

$$Jmax < p + j \qquad (1)$$

**Prova:** Se *Dmin* et *Dmax* são, respectivamente, os atrasos máximo e mínimo da transferência de uma UI do feixe B, então o jitter máximo de uma unidade de informação $u$ que pertença ao feixe $B$ pode ser expressa por:

$$Jmax = Dmax - Dmin \qquad (2)$$

O atraso mínimo associado ao fluxo guia pode ser expresso por:

$$dmin^g = Dmax - j \qquad (3)$$

De acordo com a definição da técnica de geração de *Dexpr's* da estratégia de sincronização dirigida, a unidade de informação *u* não pode ser jamais entregue antes da UI do fluxo guia que imediatamente a precede: como no caso limite esta UI é submetida um tempo *p* antes de *u*, então o atraso mínimo de *u* pode ser expresso por:

$$Dmin \geq Dmax - j - p \qquad (4)$$

Combinando as equações (2) et (4), nós obtemos a proposição (1).

**5.2.1 Sensibilidade às perdas-** A estratégia de sincronização dirigida, tal como foi definida anteriormente, implica que no caso de perda de uma UI do fluxo guia, todas as UI's dos outros fluxos do feixe submetidas entre esta UI a a próxima UI do fluxo guia também não sejam entregues. Para levar em conta este tipo de restrição, é proposta uma alternativa expressa pelo atributo *relations_number*, que oferece uma possibilidade de determinar o número de relações de dependências que serão automaticamente geradas (relações disjuntivas). Dessa forma, a técnica de geração de dependências é agora definida por:

- para cada UI não pertencendo ao fluxo guia, é criada uma expressão de dependências que a condiciona à entrega da UI do fluxo guia submetida imediatamente antes dela e, de uma forma disjuntiva, à todas as [*relations_number* - 1] unidades de informação consecutivas.

A figura 11(a) ilustra a nova configuração do feixe da figura 10(a) para *relations_number* igual *2*. Uma consequência da extensão do número de relações de dependências é o aumento do jitter residual máximo associado ao feixe. A figura 11(b) ilustra os novos valores do jitter para o feixe da figura 11(a). A generalização da inequação (1) considerando o número de relações de dependências é expressa por:

$$Jmax < p(relations\_number) + j \qquad (5)$$

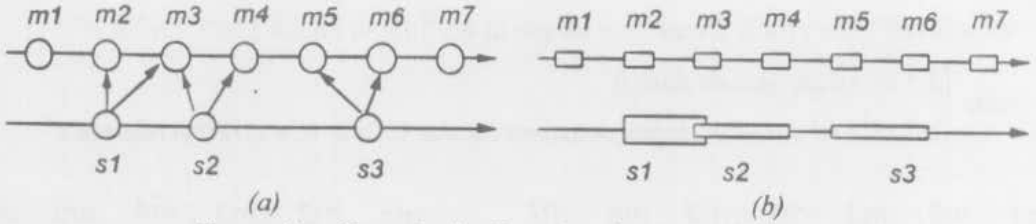


***Figura 11*** *- Geração das Dexpr's e diagrama de entrega*

Associa-se ainda mais dois atributos a transferência de um feixe pela sincronização dirigida. O parâmetro *master_conf* permite à aplicação de restringir o conjunto de UI's que será considerado durante a geração das expressões de dependências (por exemplo se *master_conf* = 4 então somente a primeira UI entre um grupo de quatro UI's consecutivas do fluxo guia será considerada na hora de gerar as *Dexpr's*). O parâmetro *active_stream_set* determina o conjunto de fluxos do feixe que deve seguir a estratégia de sincronização dirigida.

```
S_dirigida_perfil (conducted_profile)
        Attribute:      conducted_profile
                        Attribute:          stream_leader
                        Attribute:          relations_number
                        Attribute:          mastre_conf
```

**Exemplo** - Essa estratégia se adapta perfeitamente às transferências de feixes que incluem um fluxo fortemente isócrono como o caso de fluxos de vídeo. Nós escolhemos o exemplo clássico de sincronização dos lábios num feixe composto de um fluxo de audio e um fluxo de vídeo. Em [SE 93], Steinmetz desenvolveu algumas medidas relacionadas a percepção humana da sincronização multimídia e ele propôs um valor de 85 ms para a defasagem máxima entre as UI's de um fluxo de audio e as do fluxo de vídeo. Considerando que o fluxo de vídeo é composto de 25 quadros por segundo (período de 40 ms) e que ele suporta um jitter residual de 5 ms, então pode-se utilizar uma estratégia de sincronização dirigida com *relations_number* = 2, já que a aplicação da inequação (5) garante um desvio máximo entre as UI's do fluxo de vídeo e do fluxo de audio de 85 ms.

### 5.3 Sincronizacão acoplada

Esta estratégia de sincronização é uma variação da estratégia de sincronização dirigida e consiste essencialmente de trocar a geração de dependências simples pela geração de pontos de sincronização. Esse procedimento reforça os elos de sincronização entre os fluxos de um feixe.

O procedimento de geração das expressões de dependências é um pouco mais complexo, já que um ponto de sincronização expressa relações de dependências mutuais. A utilização do mesmo procedimento da *sincronização dirigida* poderia provocar situações onde seria necessário especificar uma expressão de dependência sem ter um conhecimento preciso do argumento a ser empregado. A razão disto é a possibilidade de existir uma defasagem entre as submissões das UI's que vão compor um ponto de sincronização. Essa defasagem implicaria que, durante a submissão da primeira UI, não fosse ainda conhecido o identificador da UI que seria associada ao ponto de sincronização (nem se existiria realmente um ponto de sincronização).

A estratégia de geração de pontos de sincronização resolve este tipo de problema retardando as UI's do fluxo guia na entidade emissora até a chegada da próxima UI:

i) cada UI submetida à entidade de sincronização emissora pertencente ao fluxo guia é retida até a chegada da próxima UI, antes de ser repassada aos serviços de comunicação;
ii) para cada UI do feixe, que não pertença ao fluxo guia e que tenha chegado durante o intervalo de tempo em que a UI do fluxo estiver retida, é criada uma expressão de dependências condicionando-a à UI do fluxo guia;
iii) as UI's do fluxo guia são transmitidas com uma expressão de dependências conjuntiva à todas as outras UI's (condição (ii)) que foram emitidas durante seu período de retenção.

Esses procedimentos são ilustrados pela figura 12:

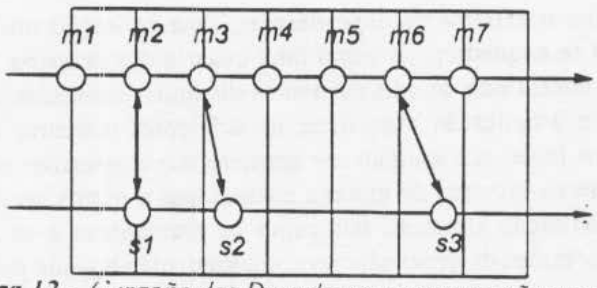


***Figura 12** - Geração das Dexpr's na sincronização acoplada*

**Definição 2:** Uma transferência de um feixe $B$ pela estratégia de sincronização acoplada deve satisfazer às seguintes condições :

$$\forall S \in (B \setminus \{S^m\}) \; e \; \forall i \in S,$$

$$(Dexpr^i = a \; e \; Dexpr^{ii} = i) \; tal \; que \; (a \in S^m) \; e \; \neg(\exists b \in S^m : ts^{ii} < ts^h \leq ts^i)$$

$$onde \begin{cases} S^m \; é \; o \; fluxo \; guia \; do \; feixe \; B \\ ts^a, ts^h \; e \; ts^i \; são \; os \; instantes \; de \; submissão \; das \; UI's \; a, \; b, \; e \; i, \; respectivamente \end{cases}$$

A aplicação da estratégia de sincronização dirigida no exemplo da figura 12 fornece um diagrama temporal de entrega conforme a figura 13. O jitter máximo deste tipo de estratégia de sincronização é calculado da mesma forma proposta para a estratégia de sincronização dirigida. Esta estratégia apresenta uma característica peculiar, expressa pela entrega acoplada das UI's do fluxo guia e das outras UI's do feixe (de acordo com a definição de ponto de sincronização). O único parâmetro que compõe o perfil da sincronização acoplada é a definição do fluxo guia (*stream_leader*).

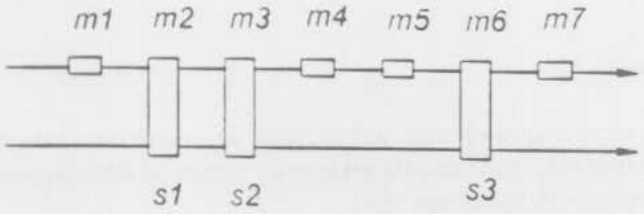


*Figura 13 - Diagrama temporal da entrega das UI's para uma sincronização acoplada*

## 6. Conclusões

Este artigo enfocou principalmente o emprego de uma nova técnica de sincronização para sistemas de comunicação baseada na identificação de relações causais, também chamadas de relações de dependências, entre as unidades de informação de um ou vários fluxos de um objeto multimídia. A especificação destas relações causais é feita através de expressões de dependências, sendo que os mecanismos de entrega condicional garantem que estas relações causais, especificadas a nível de usuário, sejam satisfeitas no instante de restituição do objeto multimídia (entrega dos fluxos à aplicação). Para cada unidade de informação recebida da camada de transporte, o mecanismo determina se esta UI deve ser entregue ou não, em função da análise dinâmica da sua expressão de dependências. Após mostramos o interesse da estratégia de sincronização baseada na avaliação das expressões de dependências condicionais, nós passamos a definição de uma ferramenta que traduza esta estratégia de sincronização sob a forma de um serviço de comunicação, em outras palavras, de como estabelecer as expressões de dependências que serão empregadas durante a transferência de um feixe multimídia.

Para isso definiu-se o extrator de dependências condicionais, que pode ser visualizado como uma ferramenta de auxílio a especificação da qualidade de serviço de sincronização. O principal objetivo do extrator de dependências condicionais é permitir um mapeamento automático de cenários de apresentações multimídias em esquemas de dependências condicionais que vão controlar a transferência das unidades de informação dos fluxos envolvidos.
Da maneira como foi definido o extrator de dependências, uma aplicação multimídia pode escolher em qual nível de abstração ela deseja se enquadrar, tal como uma usuária dos serviços de sincronização. O extrator de dependências condicionais é constituído de dois elementos distintos: *a unidade de especificação das expressões de dependências* que permite a aplicação especificar de diferentes maneiras as expressões de dependências condicionais associadas a um feixe; e *a unidade de geração das expressões de dependências* que estabelece equivalências funcionais entre os serviços de entrega condicional e outros serviços de sincronização baseados em outras técnicas de sincronização clássicas, tais como os marcadores e os canais de sincronização. Neste último caso a geração das expressões de dependências é inteiramente efetuada pelo extrator de dependências.

Os próximos passos incluem o desenvolvimento de um conjunto de facilidades para a passagem automática de um cenário de apresentação multimídia (ex: especificado através de MHEG, HyTime, etc.) para um esquema de dependências condicionais, e a criação de uma biblioteca de esquemas de relações de dependências genéricas, que irá permitir aos usuários indicar apenas o tipo de codificação dos fluxos e, a partir daí, as relações de dependências serão automaticamente deduzidas.

## 7. Bibliografia

[AH 91] D.P. Anderson and G. Homsy, Synchronization policies and mechanisms in a Continuous Media I/O Server, Report N°UCB/CSD 91/617, University of California, Berkeley (USA), February 1991.

[Carmo 95] L.F.R.C. Carmo, "Méthodes de Synchronisation dans les Systèmes répartis multimédia: une approche intégrant relations de causalité et containtes temporelles", Thèse de doctorat de l'UPS - Toulouse/FR, N. ordre 1867, Rapport LAAS 94422, 1994.

[CC 94a] L.F.R.C. Carmo, JP. Courtiat, "Implementing Intra-stream Synchronization by means of Conditional Dependency Expressions", Proceedings of the 5th IFIP Conference on High Performance Networking, France, 1994.

[CCO 94] J.P. Courtiat, L.F.R.C. Carmo, R.C. de Oliveira, "A General-purpose Multimedia Synchronization Mechanism Based on Causal Relations", LAAS report N° 94278, IEEE

Journal on Selected Areas in Communications - Synchronization Issues in Multimedia Communications, 1995.

[CCO 94] J.P. Courtiat, L.F.R.C. Carmo, R.C. de Oliveira, "A New Mechanism for Achieving Inter-stream Synchronization in Multimedia Communication Systems", Proceedings of the IEEE International Conference on Multimedia Computing and Systems (ICMCS '94), USA, 1994.

[CCS 93] L.F.R.C. Carmo, J.P. Courtiat, P. de Saqui-Sannes, "Towards Multimedia Communication Services", Proceedings of the Fourth Workshop on Future Trends of Distributed Computing Systems, Lisbon, IEEE Press, 1993.

[COC 94] J.P. Courtiat, R.C. de Oliveira, L.F.R.C. Carmo, "Towards a new multimedia synchronization mechanism and its formal specification", Proceedings of ACM Multimedia'94, USA, October, 1994

[CSC 92] L.F.R.C. Carmo, P. de Saqui-Sannes, J.P. Courtiat, "Basic Synchronisation Concepts in Multimedia Systems", Proceedings of the 3rd IEEE International Workshop on Network and Operating System Support for Digital Audio and Video, San Diego (USA), November 1992; also in Lecture Notes in Computer Sciences N°712, Springer Verlag, 1993.

[ISO 92] ISO, "Hypermedia/Time-Bases Structuring Language (HyTime)", ISO/IEC IS 10744, August, 1992.

[ISO 93a] ISO, "Coding of Moving Pictures and Associated Audio", Committee Draft for standard ISO/IEC JTC1/SC29, WG11/602, November, 1993.

[ISO 93b] ISO, "Coded Representation of Multimedia and Hypermedia Information Objects (MHEG)", Committee Draft, ISO/IEC CD 13522-1 (June 1993).

[HFK 95] N. Hirzalla, B. Falchuk, A. Karmouch, "A Temporal Model for Interactive Multimedia Scenarios", IEEE Multimedia, Fall 1995

[LIT 90] T.D.C. Little and A. Ghafoor, "Synchronisation and Storage Models for Multimedia Objects", IEEE Journal on Selected Areas in Communications, Vol. 8, N°. 3, 1990.

[LKG 94] L. Li, A. Karmouch and N.D. Georganas, "Multimedia Teleorchestra with Independent Sources: Part 1 - Temporal Modelling of Collaborative Multimedia Scenarios & Part 2 - Synchronization Algorithms, Multimedia Systems, Vol. 1, N°.4, Springer Verlag

[PAR 91] C. Partridge, "Isochronous Applications Do Not Require Jitter-Controlled Networks", Request for Comments 1257, 1991.

[RRVK 93] P. V. Rangan, S. Ramanathan, H. M. Vin adn T. Kaeppner, "Techniques for Multimedia Synchronization in Network File Systems, Computer Communications, Vol. 16, N°. 3, 1993.

[SDS 93] H. Santoso, L. Dairaine, S. Fdida and E. Horlait, "Preserving Temporal Signature: a Way to Convey Time Constrained Flows", Proceedings of IEEE GLOBECOM'93 Conference, Houston (USA), 1993.

[SE 93] R. Steinmetz, C. Engler, "Human Perceception of Media Synchronization", IBM Technicca1 report N° 43.9310, IBM European Networking Center, Germany, 1993.

[STE 90] R. Steinmetz, "Synchronization Properties in Multimedia Systems", IEEE Journal on Selected Areas in Communications, Vol. 8, N° 3, April 1990.

[WQ 94] M. Woo, N.U. Qazi and A. Ghafoor, "A Synchronization Framework for Communication of Pre-orchestrated Multimedia Information", IEEE Network, Vol. 8, N°1, January/February 1994.