Tradutor Automático de Especificações E-LOTOS para o padrão MHEG-5

André Luiz Kawamoto[1] Wanderley Lopes de Souza
PPG-CC/DC/UFSCar
Caixa Postal 676
13565-905 São Carlos - SP
e-mail: { kawamoto, desouza}@dc.ufscar.br

Resumo

No projeto *Design de Aplicações Multimídia Distribuídas (DAMD)*, financiado pelo Programa Temático Multi-institucional em Ciência da Computação (ProTeM-CC) Fase 3, foi desenvolvido um ambiente integrado de ferramentas para a Técnica de Descrição Formal (TDF) *Enhancements to Language of Temporal Ordering Specification (E-LOTOS)*, que dá suporte a todo o processo de autoria de aplicações multimídia distribuídas (especificação, validação, implementação e teste). Este artigo apresenta um tradutor automático de aplicações multimídia, realizadas em E-LOTOS, para o padrão *Multimedia and Hypermedia Information Coding Expert Group - Part 5 (MHEG-5)*.

Palavras-chave: Técnicas de Descrição Formal, Multimídia, E-LOTOS, MHEG-5

Abstract

On the *Design of Distributed Multimedia Applications project (DAMD)*, supported by *Multi-institutional Thematic Program on Computer Sciences (ProTeM-CC)* phase 3, an integrated environment of tools was developed for the Formal Description Technique (FDT) *Enhancements to Language of Temporal Ordering Specification (E-LOTOS)*, which supports the whole authoring process of distributed multimedia applications (specification, validation, implementation and testing). This paper presents an automatic translator from E-LOTOS specifications of multimedia applications to the *Multimedia and Hypermedia Information Coding Expert Group - Part 5 (MHEG-5)* standard.

Keywords: Formal Description Techniques, Multimedia, E-LOTOS, MHEG-5

1. Introdução

Recentemente, os serviços multimídia têm se mostrado um emergente campo de aplicação. O surgimento de novas tecnologias possibilita o uso de sistemas multimídia distribuídos em diversos setores da atividade humana, tais como educação, medicina, entretenimento, etc..

No desenvolvimento de sistemas multimídia costuma-se utilizar modelos informais ou semi-formais, que podem ser uma fonte de ambigüidades e não possibilitam o uso de ferramentas de análise, não garantindo assim a consistência lógica e temporal de tais sistemas. O emprego de modelos formais permite a realização de especificações livres de ambigüidades, a utilização de técnicas de verificação e simulação e a implementação (semi-)automática desses sistemas. Além disso, modelos formais permitem a construção

[1] Aluno de mestrado do PPG-CC da UFSCar – financiado pela Fapesp

de uma semântica precisa, visando a produção de especificações claras, concisas e consistentes.

Dentre os diversos modelos formais existentes, *Language of Temporal Ordering Specification (LOTOS)* [1], uma linguagem de especificação baseada em álgebra de processos que foi desenvolvida e padronizada pela *International Organization for Standardization (ISO)*, alcançou um bom nível de aceitação entre profissionais das áreas acadêmica e industrial. No entanto, LOTOS é limitada em suas características, principalmente no que diz respeito à representação da noção de tempo quantitativo e à descrição de tipos de dados de uma forma amigável. Devido a essas limitações, foi proposta pela ISO uma extensão para LOTOS, denominada *Enhancements to LOTOS (E-LOTOS)* [2].

E-LOTOS mantém a mesma estrutura de LOTOS, reservando uma parte baseada em álgebra de processos e outra para a definição de dados. A partir de LOTOS, esta TDF busca aumentar a expressividade e abstração e fornecer uma interface amigável com usuário. Dentre as melhorias da linguagem, as que causam maior impacto na semântica são: a introdução da noção de tempo quantitativo na parte comportamental; a modelagem de dados de forma a fornecer uma interface mais amigável; definição de dados concisa.

Com o advento de novos protocolos de comunicação, surgiu a necessidade de um padrão para dar suporte ao processamento e troca de informações multimídia em sistemas abertos. Neste sentido, foram concentrados esforços pela ISO e pela International Eletrotechnical Comission (IEC) para o desenvolvimento da norma internacional *Multimedia and Hypermedia Information Coding Expert Group - Part 5 (MHEG-5)* [3]. Esse padrão tem como principal premissa garantir a interoperabilidade de aplicações multimídia interativas sobre plataformas heterogêneas.

O objetivo deste trabalho é apresentar um Tradutor Automático de especificações E-LOTOS para o padrão MHEG-5. Este tradutor faz parte do conjunto de ferramentas desenvolvidas no projeto "*Design de Aplicações Multimídia Distribuídas*" (DAMD) [4], financiado pelo Programa Temático Multi-institucional em Ciência da Computação (ProTeM-CC) do CNPq Fase III . Alguns dos objetivos do Projeto DAMD são a concepção, a descrição e a utilização de uma metodologia, que permita a especificação de aplicações multimídia utilizando-se a TDF E-LOTOS.

Este trabalho está organizado da seguinte forma: a seção 2 mostra o estilo de especificação para aplicações multimídia adotado no DAMD e descreve a aplicação exemplo utilizada para mostrar o processo de tradução; a seção 3 introduz o padrão MHEG-5, que é a representação final do processo de tradução; a seção 4 apresenta a metodologia de tradução em que foi baseada a implementação do tradutor; a seção 5 detalha a implementação do tradutor; finalmente, na seção 6 são apresentadas as considerações finais e propostas de trabalhos futuros.

2. Estilo de especificação de aplicações multimídia em E-LOTOS

Para que especificações E-LOTOS sejam traduzidas para MHEG-5 automaticamente, é necessário que, durante o processo de autoria de tais especificações, seja seguido um

estilo preestabelecido. Para tanto, foi proposto no DAMD um estilo [5] baseado em bibliotecas de *Restrições* e *Apresentações*.

A biblioteca de *Apresentações* é constituída de vários processos E-LOTOS que, individualmente, representam um objeto de mídia básico que corresponde a um vídeo, uma imagem, uma seqüência de áudio ou um objeto interativo (botão). Tais objetos são tratados como uma unidade lógica e são propostos para modelar cenários multimídia interativos.

Os processos dessa biblioteca denotam o comportamento temporal das mídias através de seus eventos de início e fim. A combinação de eventos síncronos (início e fim da apresentação) e eventos assíncronos (interação do usuário) produz vários padrões de comportamento [6]: *Synchronous Start and Synchronous End* (pSsSe), *Synchronous Start and Asynchronous End* (pSsAe), *Synchronous Start and Asynchronous Maximal End* (pSsAme), *Asynchronous Start and Asynchronous maximal End* (pAsAme), *Asynchronous Start and Asynchronous End* (pAsAe), *Asynchronous maximal Start and Asynchronous End* (pAmsAe), *Asynchronous maximal Start and Synchronous End* (pAmsSe), *Asynchronous maximal Start and Asynchronous maximal End* (pAmsAme). Além desses padrões, o estilo proposto no DAMD introduz os processos pSsSeP, pSsAeP, pSsAmeP, pAsAmeP, pAsAeP, pAmsAeP, pAmsSeP, pAmsAmeP e pButton [7] para representar objetos interativos (Botões), e o controle de apresentação de mídias (Play, pause, stop).

A biblioteca de *Restrições* é constituída de processos E-LOTOS que definem restrições temporais para expressar inter-relações entre mídias. Esses processos são utilizados quando a sincronização não pode ser expressa facilmente. A modelagem destas restrições expressa a aplicabilidade de E-LOTOS para especificar a sincronização temporal e lógica de documentos multimídia/hipermídia.

Uma especificação de uma aplicação multimídia em E-LOTOS, de acordo com o estilo adotado no DAMD, deve ser da seguinte forma:

- a aplicação em si é a própria especificação;
- todas as cenas da aplicação são processos declarados em um módulo chamado *Cenas;*
- todas as mídias presentes na aplicação são processos do módulo *Componentes* e têm seu comportamento ditado através da instanciação de algum processo da *Biblioteca de Apresentações* (definida pelo estilo);
- todas as composições de mídias são processos declarados no módulo *Restrições* e seguem um dos modelos estabelecidos (*cWaitMaster*, *cWaitEarliest*, *cWaitLatest*, ou Sequencial).

Além disso, esse estilo determina um subconjunto de operadores E-LOTOS básico para a especificação de aplicações multimídia. Para descrever todo o processo de tradução, será utilizada uma aplicação exemplo simples.

2.1. Exemplo de uma aplicação multimídia

A aplicação exemplo é composta de duas cenas. Na primeira são apresentadas, simultaneamente, as seguintes informações:

- um áudio (Au1);
- uma imagem (Img1);
- (3) um botão (Quit), que interrompe a apresentação das demais informações no momento de sua seleção, ou decorridos 35 segundos após a sua apresentação.

Após a seleção do botão, ou do término de sua apresentação, é disparada a apresentação de uma segunda imagem (Imagem2). Essa imagem finaliza a cena dispara uma transição para outra cena.

As Figuras 1a e 1b mostram o *esqueleto* da especificação E-LOTOS dessa aplicação exemplo, que utiliza o estilo proposto no DAMD.

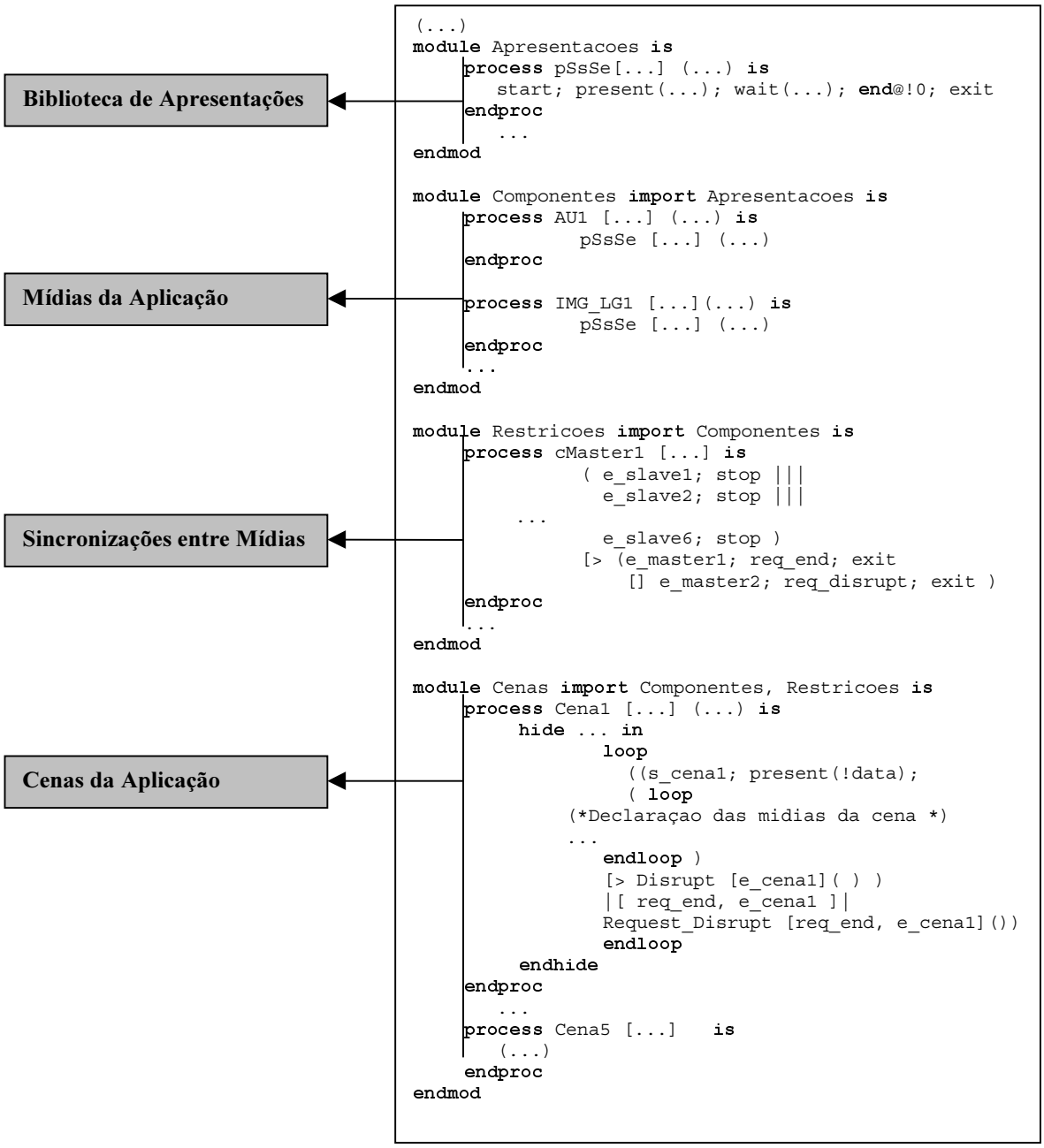

Figura 1a: Módulos da especificação representativos para a obtençao da RI

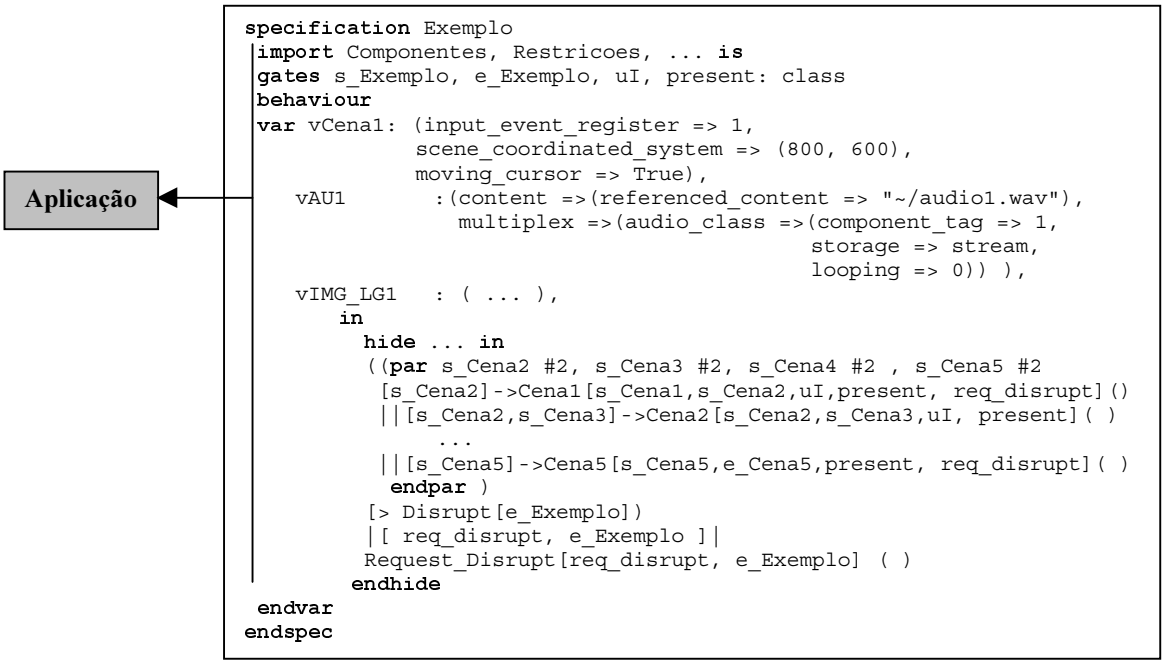

Figura 1b: '*specification'* de acordo com o estilo DAMD

3. O Padrão MHEG-5

O padrão MHEG-5 foi desenvolvido para dar suporte à distribuição de aplicações multimídia interativas em uma arquitetura cliente/servidor através de diferentes plataformas, isto é, MHEG-5 estabelece uma representação fina para aplicações multimídia/hipermídia intercambiáveis sobre plataformas heterogêneas.

Uma aplicação MHEG-5 é definida como uma hierarquia de classes baseada em uma abordagem orientada em objetos. Basicamente, uma aplicação é estruturada por cenas e objetos. Dentro de uma cena existem objetos a serem apresentados e seu comportamento é baseado em eventos. A navegação dentro de uma aplicação é feita pela composição de transições de cenas.

De acordo com Gopal et al [6a], MHEG-5 tem como sua maior funcionalidade o estabelecimento do comportamento genérico de um conjunto de classes que assegura:

- o suporte à representação final das aplicações - sua representação é suficiente para interpretação e apresentação por um *engine* MHEG-5;
- o suporte a recursos mínimos - devido à sua flexibilidade, é assegurado que ambientes com recursos mínimos apresentem facilmente aplicações intercambiáveis;
- o suporte a interações e sincronização - uma importante característica é a capacidade de descrever o resultado de interações do usuário, e habilitar a apresentação de várias mídias simultaneamente de uma maneira coordenada;
- o suporte a apresentação em tempo-real e intercâmbio de aplicações.

3.1 A Hierarquia de Classes MHEG-5

A principais classes MHEG-5 [6] envolvidas em um processo de autoria de uma aplicação são:

Root **-** esta é classe base para todas as outras classes MHEG-5. Sua principal função é fornecer semântica de comportamento genérico para estes objetos (e.g., ativação, desativação, preparação e destruição), e fornecer um mecanismo de identificação.

Group **-** A classe *Group* é classe base para as classes *Application* e *Scene*. Sua função é permitir o agrupamento de objetos que irão compor a aplicação ou a cena. Os objetos agrupados por esta classe são objetos *Ingredient*.

Application - objetos *Application* definem a aplicação em si, desta forma, apenas um objeto *Application* por vez pode estar ativo. Um engine MHEG-5 ocioso inicia uma aplicação através da preparação e ativação do objeto *Application* correspondente. Quando a aplicação se torna ativa, ela automaticamente executa uma declaração *OnStartUp* que pode ser usado para executar o primeiro objeto *Scene* da aplicação. Particularmente, qualquer objeto *Ingredient* incluído em um objeto *Application* será compartilhado pelas cenas da aplicação.

Scene - A proposta da classe *Scene* é permitir à apresentação de um cenário multimídia usar coordenadas espaciais e/ou temporais.

Ingredient **-** a classe *Ingredient* é uma classe base para as classes *Link*, *Program*, *Palette*, *Font*, *CursorShape*, *Variable* e *Presentable*. Esta classe determina o comportamento genérico dos objetos que compõem um *Scene* ou *Application*. Estes objetos contêm informações que serão apresentadas ao usuário (classes *Pallette*, *Font*, *CursorShape*, *Variable* e *Presentable*) e características do comportamento da aplicação (classes *Link* e *Program*).

Link **-** objetos *Link* são usados para expressar o comportamento de aplicações MHEG-5. Um objeto *Link* consiste de uma condição e um objeto *Action* (*LinkEffect*). Quando a condição é verdadeira, o *Link* dispara a execução dos objetos *Action* associados.

Action - a função de um objeto *Action* é executar, em uma seqüência síncrona, uma série de ações elementares como resultado do *Link* disparado. Uma ação elementar pode mudar o estado de objetos, criar e destruir objetos, ou gerar outros eventos.

Presentable: a classe *Presentable* é uma classe base para as classes *Audio*, *Visible*, *TokenGroup* e *Stream*. Objetos *Presentable* representam informação que pode ser diretamente percebida pelo usuário. Esta classe permite o manuseio (opcional) da representação de codificação de conteúdo de dados corrente. Isso pode ser feito por inclusão ou referência. Por inclusão, o conteúdo de dados é transferido como parte do objeto *Presentable*; por referência, o objeto *Presentable* apenas fornece uma referência externa para os dados.

4. Metodologia de Tradução E-LOTOS/MHEG-5

Baseada no estilo de especificação adotado no DAMD, foi proposta uma metodologia para a tradução E-LOTOS/MHEG-5 [8] constituída de duas fases distintas: (1) obtenção de uma Representação Interna (RI) a partir da especificação E-LOTOS; (2) tradução da RI obtida para MHEG-5. A RI diminui a heterogeneidade existente entre E-LOTOS e

MHEG-5 e simplifica a implementação do Tradutor automático, possibilitando a representação dos componentes da aplicação multimídia, suas relações lógicas e temporais, e suas características de apresentação e conteúdo.

Figura 2. A metodologia para a tradução E-LOTOS/MHEG-5

A Representação Interna de uma aplicação multimídia é resultado da composição de quatro componentes:

- *Application*, que representa a aplicação multimídia de modo geral. Este componente é constituído por todas as cenas da aplicação e pela declaração da primeira cena a ser apresentada;
- *Scene*, que representa os cenários da aplicação. Neste componente são declaradas suas mídias, suas composições de mídias, seus atributos (e.g., características de apresentação) e sua composição lógica e temporal;
- *Composition,* que representa um agrupamento de mídias associadas por uma sincronização lógica e/ou temporal. Neste componente são declaradas suas mídias, outras possíveis composições de mídias e seus atributos;
- *Presentation,* que representa as mídias a serem apresentadas, assim como os objetos interativos (e.g., botões) dos cenários.

Para a obtenção dos componentes da RI, foi desenvolvida uma estratégia a partir da especificação E-LOTOS. Percorre-se toda a especificação identificando as cenas (processos do módulo *Cenas*), apresentações (processos que instanciam processos da biblioteca de *Apresentações*), composições (processos que instanciam processos da biblioteca de *Restrições*). As características de apresentação da aplicação e das mídias que a compõem são obtidas a partir da declaração de variáveis. A primeira cena a ser apresentada é obtida a partir da semântica do operador de paralelismo genérico de E-LOTOS.

Após a obtenção da RI, a segunda etapa da metodologia é o mapeamento para MHEG-5. Uma aplicação MHEG-5 é descrita através da declaração de dois objetos:

- *Application*, que define a aplicação a ser apresentada;
- *Scene*, que caracteriza os cenários multimídia da aplicação e seus componentes.

Para que uma aplicação MHEG-5 possa ser apresentada, ela deve ser composta por apenas um objeto *Application* e pelo menos um objeto *Scene*. Assim sendo, a tradução da RI para MHEG-5 é orientada à geração dos objetos MHEG-5 *Application* e *Scene*.

O objeto MHEG-5 *Application* é traduzido a partir da declaração do componente *Application* da RI, onde são declarados o nome da aplicação, seus componentes (cenas), e a primeira cena a ser apresentada (atributo *start*) (Figura 3).

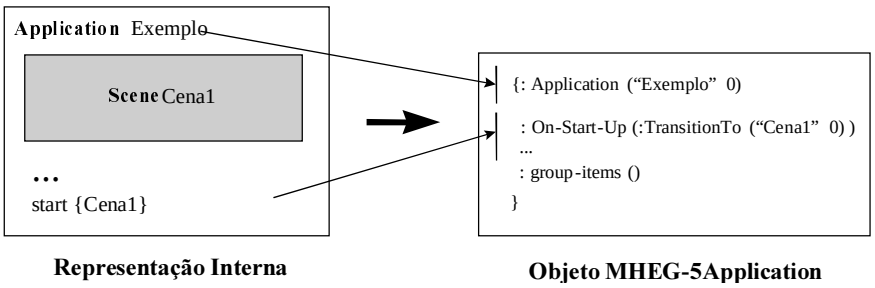

Figura 3 - Tradução RI->MHEG-5 do objeto *Application*

O objeto MHEG-5 *Scene* é traduzido a partir dos componentes *Scene* da RI (Figura 4). No processo de tradução, o mapeamento das cenas da aplicação é um-a-um, procurando-se representar em MHEG-5 o nome da cena, as mídias a serem apresentadas inicialmente, seus componentes (composições e mídias), a composição lógica e temporal da cena e suas características de apresentação.

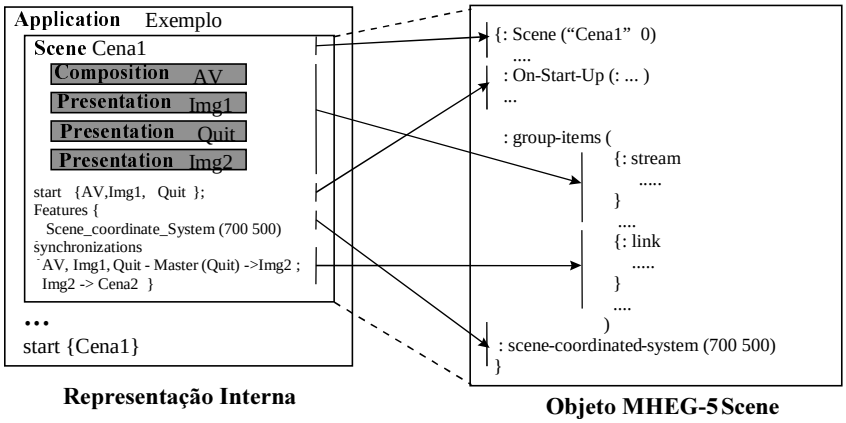

Figura 4 - Tradução RI->MHEG-5 do objeto *Scene*

O nome da cena é traduzido a partir do atributo *name* da declaração do componente *Scena*. As informações (mídias), a serem apresentadas inicialmente, são traduzidas a partir do atributo *start* da *Cena* e devem ser declaradas no atributo *On-Start-Up* do objeto *Scene* para a sua ativação.

As mídias e as composições de mídias são traduzidas, respectivamente, a partir dos componentes *Presentation* e *Composition* de *Scene*. As mídias a serem apresentadas na cena, assim como suas composições, são declaradas no atributo *group_items* do objeto *Scene*.

O comportamento da cena, ou seja, sua composição lógica e/ou temporal é expresso pelo atributo *synchronizations* na RI. Esse atributo é constituído de expressões que denotam a composição seqüencial ou paralela dos componentes da cena. Cada expressão de sincronização é traduzida nos objetos *links* declarados no atributo *group-items*.

5. Implementação do Tradutor

A implementação do tradutor automático foi realizada em duas fases, de acordo com a metodologia proposta. A primeira fase da metodologia visa a implementação de um tradutor que gera uma RI a partir da especificação E-LOTOS (Figura 5).

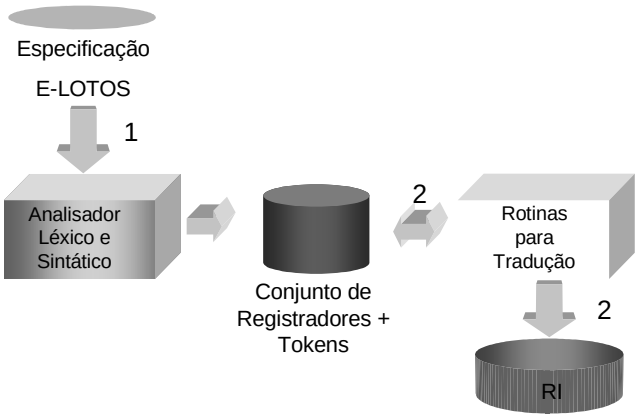

Figura 5. Implementação do Tradutor E-LOTOS/RI

Primeiramente é realizada uma análise léxica e sintática sobre a especificação. Para a construção do analisador foi utilizada a ferramenta *Purdue Compiler Construction Tool Set (PCCTS)*[9], que constrói um analisador léxico e sintático a partir da gramática de E-LOTOS, em linguagem de programação C++ orientada a objetos.

A análise léxica identifica os tokens da linguagem, enquanto a análise sintática verifica se a especificação está de acordo com o estilo proposto, ou seja, apenas o subconjunto de operadores E-LOTOS básico deve ser utilizado na especificação.

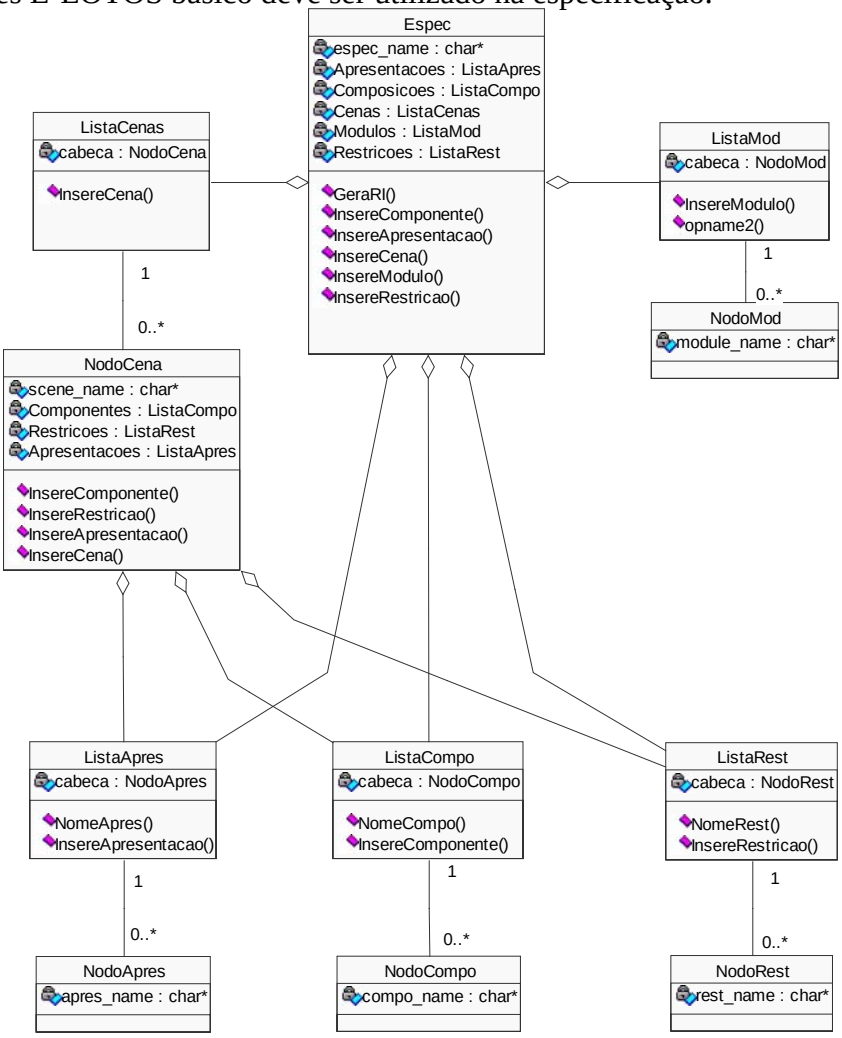

Figura 6: Diagrama de Classes utilizado na implementação do Tradutor E-LOTOS/RI

Durante a análise, os dados da especificação são armazenados em objetos. Esses objetos foram implementados para permitir a análise semântica da especificação, e visando a geração da RI. Os objetos relevantes para a criação da RI são (Figura 6):

- *Espec*, que guarda os dados referentes à especificação em si, como por exemplo o nome da aplicação, as listas de apresentações, composições e cenas presentes;
- *Apresentações*, que é uma lista de todos os processos presentes na biblioteca de Apresentações;
- *Componentes*, que é uma lista de todos os processos declarados que representam mídias a serem apresentadas nas cenas da aplicação.
- *Composições*, que é uma lista de todos os processos do módulo *Restrições,* os quais representam todas as composições lógicas e temporais que existem entre mídias da aplicação.

Ao final da análise, se a especificação estiver correta e de acordo com o estilo adotado no DAMD, os dados contidos nos objetos são utilizados pelo método *GeraRI* do objeto *Espec*, sendo que é gerado o arquivo texto com a declaração da RI. A Figura 8 mostra a RI obtida a partir da especificação exemplo.

```
application Exemplo
  { scene Cena1
    {composition cMaster1
      { presentation AU1
      { model pSsSe;
        type StreamClass;
        features
        { InitiallyActive False;
          Content
          ContentReference ( " ~/audio1.wav"; );
          Multiplex ( }; };
        presentation IMG_LG1
      { model pSsSe;
        type BitmapClass;
        features
        { InitiallyActive False;
          Content
          ContentReference ( " ~/img_lg1.bmp"; );
          OriginalBoxSize 768 281;
          OriginalPosition -10 98; }; };
        presentation IMG1
      { model pSsSe;
        type BitmapClass;
        features
        { InitiallyActive False;
          Content
          ContentReference ( " ~/img1.bmp"; );
          OriginalBoxSize 768 281;
          OriginalPosition -10 98; }; };
...
start {  AU1 IMG_LG1 IMG1 TX1 IN1 IMG_W1 bInicializar bFim }
         syncronizations
           { AU1, IMG_LG1,..., bInicializar, bFim - cWaitMaster(
bInicializar bFim ) ->transition_to Cena2 }; };
   start {  AU1 IMG_LG1 ... bInicializar bFim}
   features
  {
    InputEventRegister 1;
SceneCoordinateSystem 800 600;
    MovingCursor True;
  };
   scene Cena5
    {...};
  start { Cena1 };
 };
```

Figura 8 : RI da aplicação exemplo

A segunda fase da implementação concentrou-se no Tradutor RI/MHEG-5. Foi construído um analisador léxico para a identificação dos componentes da RI (*Application*, *Scenes*, *Composition* e *Presentation*), utilizando a ferramenta *PCCTS*. A Figura 9 ilustra de maneira geral o processo de tradução a partir da RI, e a Figura 10 apresenta o diagrama de objetos criados para essa implementação.

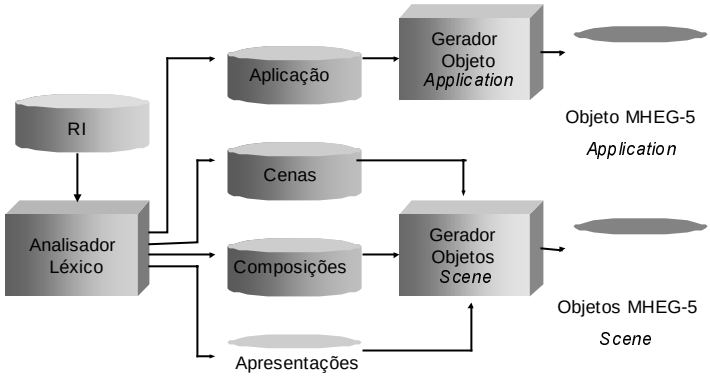

Figura 9. Implementação do Tradutor RI/MHEG-5

Os dados referentes à cada componentes identificado pelo analisador são armazenados nos objetos e utilizados pelos métodos *GeraMHEGScene* e *GeraMHEGApplication* para criar os arquivos textuais contendo a notação dos objetos MHEG-5 *Application* e *Scene*. Os métodos implementados nessa fase são detalhados em [8].

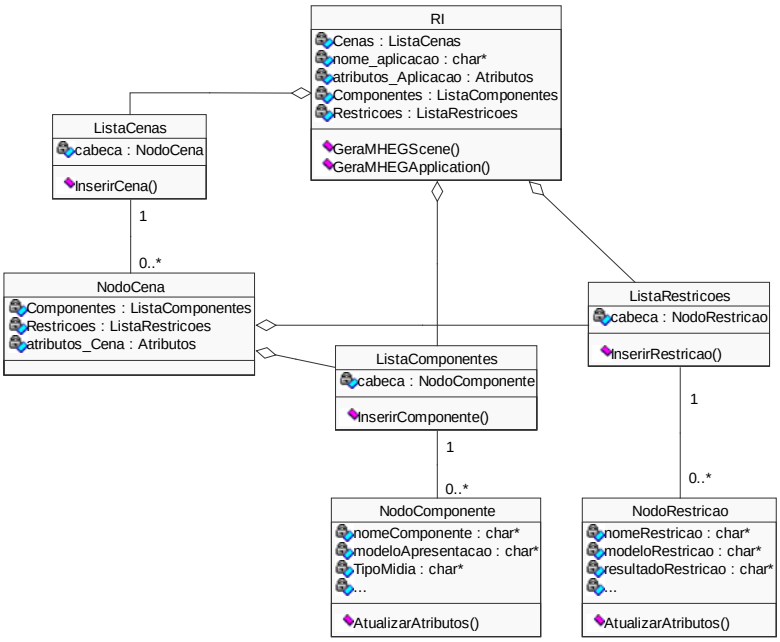

Figura 10: Diagrama de Classes utilizado na implementação do Tradutor RI/MHEG-5

O resultado final de todo o processo de tradução são os objetos MHEG-5 que poderão ser intercambiados e apresentados por uma máquina MHEG-5. As figuras 11a e 11b mostram, respectivamente os objetos MHEG-5 *Application* e *Scene gerados* a partir da especificação exemplo.

```
{       :Application
        :object-identifier (’Exemplo’ 0
        :on-start-up (
                        :transition-to (’Cena1’ 0)
                    )
        :Items()
}
```

Figura 11a : Objeto MHEG-5 *Application*

```
{:scene
 :object-identifier ("Cena1" 0)
 :group items (
        {:stream
                :object-identifier 1
                :initially-active true
                :content-data :referenced-content " ~/audio1.wav"
                :multiplex ( ( 1
        {:audio
                :object-identifier 1011
                :initially-active true
                   } ) )
                :storage
                :looping
                }
        {:bitmap
                :object-identifier 2
                :initially-active true
                :content-data :referenced-content " ~/img_lg1.bmp"
                :original-box-size (768 281)
                :original-position (-10 98)
                }
...
                    )
        :input-event-register 1
        :scene-coordinate-system (800 600)
        :aspect-ratio ( )
        :moving-cursor True
        :next-scenes
        }
}
```

Figura 11b : Objeto MHEG *Scene*

6. Considerações Finais e Trabalhos Futuros

Foi apresentada a implementação de um tradutor automático, baseada na metodologia para tradução E-LOTOS/MHEG-5 desenvolvida no projeto DAMD. Nessa implementação foi utilizada a linguagem de programação C++, e a ferramenta foi compilada para as plataformas Windows/UNIX.

Como trabalhos futuros, esse tradutor deverá ser integrado às ferramentas de edição desenvolvidas no projeto DAMD (Editor Gráfico E-LOTOS – E-DART [10] e Editor de Linguagem de Autoria [11]). Além disso, estão previstos testes com o tradutor utilizando especificações de aplicações distribuídas de maior porte, como por exemplo aplicações de Video-sob-Demanda.

Com base na experiência adquirida percebeu-se que a RI atual, por ser baseada em MHEG-5, é mais próxima a esse padrão do que à TDF E-LOTOS. Como conseqüência, toda extensão ao estilo de especificação adotado implica em alterações na RI. Devido a isso, está sendo desenvolvida uma nova RI ,mais próxima à especificação, capaz de representar a especificação com simplicidade, e de ser fator de integração ao conjunto de ferramentas do DAMD.

Referências Bibliográficas

[1] ISO/IEC DP 8807, LOTOS - “A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour”, 1989.

[2] ISO/IEC JTC1/SC21/WG7, “Enhancements to LOTOS”, Final Commite Drafts on Enhancements to LOTOS , September, 1997.

[3] ISO/IEC IS 13522-5, “Information Technology - Coding of Multimedia and Hypermedia Information : Part 5 : Support for Base-Level Interactive Applications”,1996.

[4] W. Lopes de Souza, J.-M. Farines, M.J.B. Janilce, L.F. Pires, M.S. Camargo, R. Willrich, R.J.C. Costa. "Design de Aplicações Multimídia Distribuídas (DAMD) ". Anais do II Seminário Franco-Brasileiro em Sistemas Informáticos Distribuídos, pp. 271-281, 03-07/nov/97, Fortaleza (CE).

[5] C. Y. Shiga, “Uma abordagem para especificação E-LOTOS de aplicações multimídia interativas”, Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação (PPG-CC) – UFSCar, Agosto, 1998.

[6] J.-P.Courtiat & R.C.de Oliveira, «*Proving Temporal Consistency in a New Multimedia Synchronization Model*», ACM Multimedia, Boston, MA, USA, 1996.

[7] R.M.Scheffel, P.N.M.Sampaio, C.Y.Shiga & R.Willrich. «*Especificando Aplicações Multimídia Interativas em E-LOTOS*». Nota Interna - Projeto DAMD, Outubro, 1997.

[8] P.N.M.Sampaio, “Uma Metodologia para a Implementação em MHEG-5 de Aplicações Multimídia Interativas Especificadas E-LOTOS”, Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação (PPG-CC) - UFSCar, Fevereiro, 1998.

[9] T. J. Parr, “*Language Translation Using PCCTS & C++”.* Automata Publishing Company ISBN: 0-9627488-5-4, 1998

[10] L.Z. Graville, M.B. Almeida. E-DART: Um Ambiente para Especificação de Sistemas E-LOTOS . Anais do SBRC - 16º Simpósio Brasileiro de Redes de Computadores. Rio de Janeiro, Brasil, 25-28 de maio de 1998

[11] L.P. Gaspary, M.B. Almeida. MUSE - Um Ambiente para Modelagem de Aplicações Multimídia Interativas com Tradutor para E-LOTOS. Anais do SBRC'98 - 16º Simpósio Brasileiro de Redes de Computadores. Rio de Janeiro, Brasil, 25-28 de maio de 1998.