2
Conceitos Básicos

Este capítulo tem como principal objetivo apresentar os conceitos que servem de sustentação para as discussões realizada nos capítulos seguintes. Em especial, os conceitos de grafo de cena e grafo de rotas são discutidos como abstrações declarativas para a autoria de aplicações 3D existentes. O NCM (*Nested Context Model*) e a NCL (*Nested Context Language*) também são apresentados, com foco nos pontos que devem ser estendidos ou modificados por esta dissertação.

2.1.Grafo de Cena

Segundo (PEREIRA e EBRAHIMI, 2002) uma cena é tudo que se pode ver e ouvir em uma apresentação tridimensional. Esse conceito, entretanto, é restrito na medida em que não considera que outros tipos de mídia também podem estar presentes na cena (tais como mídias olfativas, gustativas ou táteis). De uma forma mais geral, entretanto, como é apresentado em (HENDERSON e HOLLINGWORTH, 1999) uma cena é tipicamente definida como uma visão do ambiente real (frequentemente nomeável) *semanticamente coerente*, geralmente composta por objetos de segundo plano (*background*) e múltiplos objetos discretos organizados espacialmente. Embora esses objetos discretos não necessariamente necessitem ter uma representação visual (ou apenas uma representação visual) esse é o caso mais comum, daí a definição simplificada dada por (PEREIRA e EBRAHIMI, 2002).

Grafos de cena são estruturas de dados usadas para organizar e gerenciar o conteúdo de dados de uma cena, tanto cenas 2D como 3D. Grafos de cena são compostos de nós e arcos (ou arestas) entre nós. Os nós representam os objetos na cena, os quais são conectados por arcos, informando que existe algum tipo de relacionamento entre esses nós conectados. Cada nó possui uma lista de atributos, que possuem informações de como ele deve ser exibido. Adicionalmente, também é possível que alguns desses atributos influenciem seus nós conectados. Os nós e

PUC-Rio - Certificação Digital Nº 0821387/CA

arcos produzem uma estrutura em grafo que organiza os objetos que compõem uma cena de forma hierárquica (WALSH, 2002).

A estrutura utilizada como base na programação orientada a grafo de cena pode ser tanto um grafo direcionado acíclico (DAGs) como uma árvore. Um DAG é um grafo direcionado – os arcos são pares ordenados, ou seja, (*u*, *v*) é diferente de (*v*, *u*) – que não possui ciclo. Não possuir ciclo, significa que não existe um caminho, partindo-se de um nó, e seguindo-se uma sequência de arestas que eventualmente volte a esse nó. Uma árvore é um tipo especial de DAG onde cada nó tem apenas um pai.

Em um DAG, e consequentemente em uma árvore, existe naturalmente uma relação hierárquica entre os nós. Contudo, utilizar um DAG ao invés de uma árvore para modelar um grafo de cena traz a vantagem de permitir o compartilhamento de nós (um nó pode ter mais de um pai) evitando-se o aumento da complexidade do código e o consumo desnecessário de memória (repetindo-se a especificação de um nó idêntico). A Figura 1, a seguir, apresenta um exemplo de um grafo de cena baseado em DAG. Nela, é possível observar que o nó *Geometria3* possui dois pais: *Agrupamento2* e *Agrupamento3*. Em um grafo de cena baseado em árvore, esse recurso não é possível.

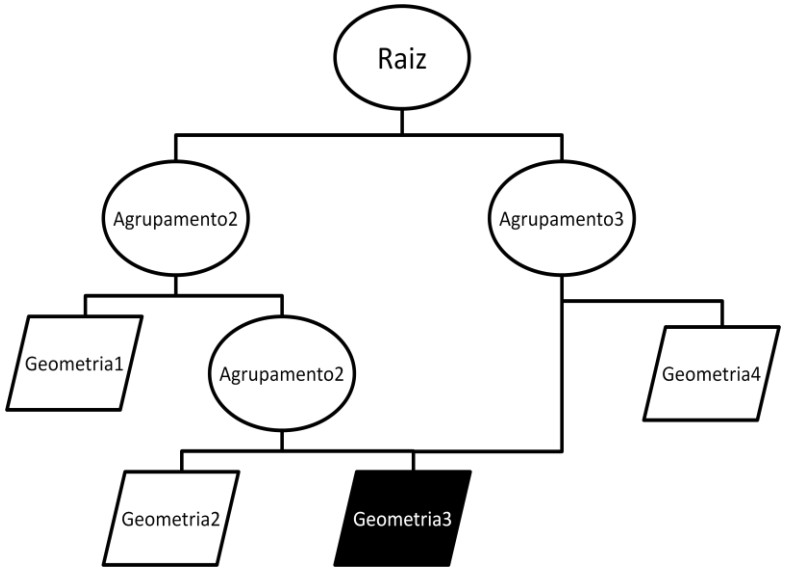

Figura 1 Modelo de um grafo de cena baseado em DAG.

Para modelar conteúdos tridimensionais, os grafos de cena definem: nós que representam primitivas geométricas – cubos, cones, esferas etc. – ou que permitem definir polígonos, ambos denominados de *geometrias*; nós de *agrupamento*, utilizados para organizar estruturalmente a cena; além de nós de

PUC-Rio - Certificação Digital Nº 0821387/CA

iluminação, aparência, material etc., que definem como as geometrias serão apresentadas (com qual cor, textura etc.).

No que se refere à organização dos objetos da cena, a noção de agrupamento espacial é essencial para os grafos de cena. Isso se deve em grande parte às otimizações possibilitadas por essa estrutura no momento de renderizar a cena – discutidas na Subseção 2.1.1. Dessa forma, criar um grafo de cena eficiente é organizar os objetos hierarquicamente, correspondendo semântica e espacialmente ao universo modelado (WALSH, 2002).

A Figura 2 apresenta dois exemplos de modelagem para um grafo de cena. Em (a), os nós são agrupadas sem considerar seu posicionamento espacial, mas alguma outra semântica que o autor deseja – no caso, cidades são agrupadas em uma mesma estrutura, estados em outra, e países em outra. Tal modelagem não considera, por exemplo, que uma cidade possa estar no território compreendido por um estado, ou seja, uma relação espacial. Em (b) essa relação espacial é considerada. Sendo assim, do ponto de vista das otimizações possibilitadas pelo grafo de cena, pode-se afirmar que (a) é menos eficiente que (b).

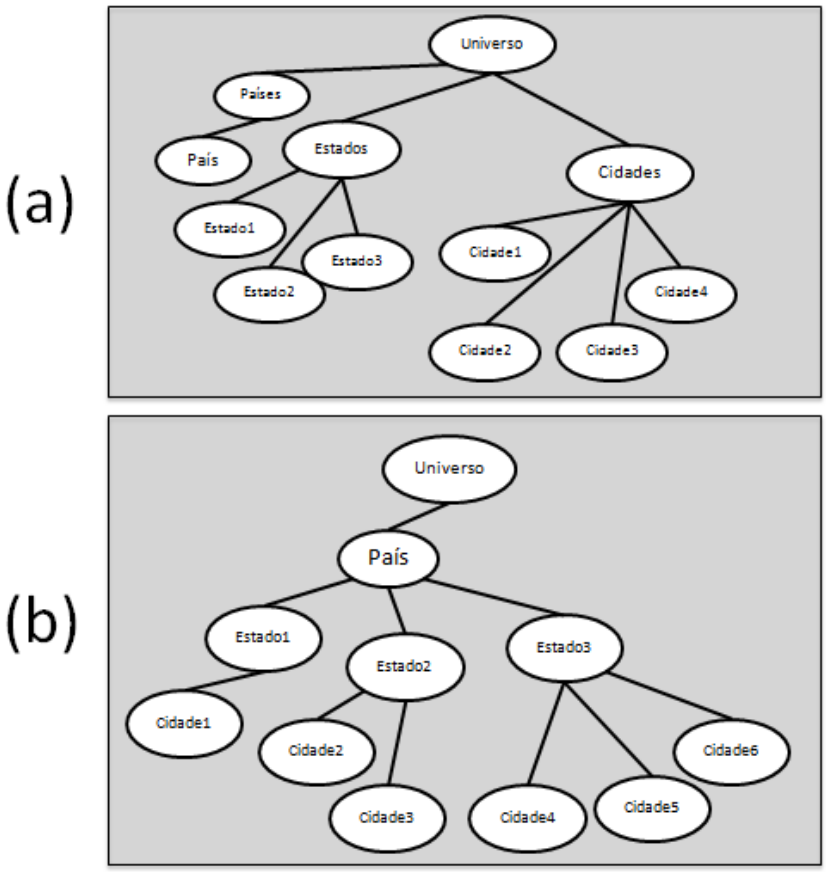

Figura 2 Grafos de cena modelados sem considerar a posição espacial (a) e considerando a posição espacial (b) dos seus nós.

PUC-Rio - Certificação Digital Nº 0821387/CA

Modelos de programação baseados em grafos de cena também podem suportar objetos com conteúdo de áudio e vídeo, temporizadores, camadas, controles de mídia, efeitos especiais e outras funcionalidades para composição multimídia (WALSH, 2002). Contudo, utilizar-se de tais composições que possuem semântica fixa e implícita na hierarquia do documento traz o inconveniente de não isolar a estrutura hierárquica de uma cena da semântica de apresentação, diminuindo assim a possibilidade de reúso dos objetos ou do comportamento de forma separada (SOARES NETO e SOARES, 2009).

No geral, a programação por meio de grafos de cena permite que o autor foque em o que será renderizado, sem ater-se em como esse conteúdo será renderizado. Isso permite que os programadores alterem seu foco de atenção de triângulos, vértices e primitivas gráficas, para começarem a pensar em objetos e sua organização na cena (SOWIZRAL, 2000). Sendo assim, percebe-se que os grafos de cena possibilitam uma menor curva de aprendizado para um desenvolvedor de aplicações, quando comparado às linguagens de programação imperativas, onde o autor deve informar, passo a passo, como a aplicação deve ser apresentada. Certamente, uma menor curva de aprendizado resulta em um aumento significativo na quantidade de potenciais desenvolvedores de aplicações 3D. Mais ainda, o grafo de cena também possibilita uma maior independência em relação ao *hardware* onde a aplicação será apresentada, já que o autor programa em alto nível e não por meio de primitivas gráficas específicas de um determinado *hardware*.

PUC-Rio - Certificação Digital Nº 0821387/CA

2.1.1.Otimizações para grafos de cena

Para renderizar um grafo de cena, uma máquina de exibição deve constantemente percorrê-lo e gerar primitivas gráficas – usualmente em OpenGL (SHREINER, WOO, *et al.*, 2005) – que devem ser enviadas para o dispositivo gráfico. A estruturação da cena, possibilitada pelo grafo de cena, entretanto permite que, muitas vezes, algumas dessas primitivas não necessitem ser enviadas para o dispositivo gráfico, seja porque um objeto não está no campo de visão do usuário, está oculto por outro objeto, ou mesmo porque está muito longe e pode ser desenhado com um menor nível de detalhe.

Não está no escopo desta dissertação discutir em detalhes todas as otimizações proporcionadas pelos grafos de cena, no momento da renderização de uma cena. Por outro lado, nesta seção são discutidas apenas as otimizações que permitem uma melhor comparação entre grafos de cena e o NCM. São exemplos das otimizações proporcionadas por grafo de cena: descartes por volume de visão (*frustum culling*), descartes por oclusão (*occlusion culling*), descarte de objetos pequenos, nível de detalhe (LOD), entre outros (SOWIZRAL, 2000).

Grande parte dessas otimizações baseiam-se no conceito de *volume envolvente* (ou *Bound Box*). O volume envolvente é uma estrutura (geralmente um paralelepípedo, um cubo ou uma esfera) que engloba o conteúdo de todos os nós abaixo de um determinado nó do grafo de cena (REINERS, 2002). Tal estrutura deve ser mantida para todos os nós de geometria ou agrupamento do grafo de cena. Em (SILVA, 2002) é apresentado um quadro comparativo entre as várias estruturas utilizadas para volume envolvente.

No *descarte por volume de visão*, por exemplo, o exibidor, ao percorrer o grafo, testa a interseção do volume envolvente com o volume de visão do observador. Se o volume envolvente estiver completamente fora do volume de visão, esse nó e todos os seus filhos, recursivamente, podem ser ignorados, não necessitando percorrê-los ou gerar as primitivas gráficas relacionadas a eles.

Assim como a interseção do volume de visão com o volume envolvente, outra operação comum enquanto uma cena está sendo apresentada é a de interseção entre volumes envolventes. Essa operação permite saber, por exemplo, se dois objetos estão colidindo. É fácil perceber que em uma organização de grafo de cena *ideal*, essa operação só precisa realmente ser calculada quando os dois nós tem o mesmo pai, caso contrário é direto afirmar que os nós não colidem, tornando-se bastante trivial.

Quanto maior a distância do observador para um objeto, menor esse objeto é apresentado na tela e, consequentemente, menos detalhes desse objeto serão visíveis por esse observador. Tentando tirar proveito disso, grande parte das implementações de grafos de cena define nós de agrupamento especiais, geralmente denominados LOD, que permitem definir representações alternativas para um mesmo objeto gráfico. A escolha de qual das representações alternativas será exibida em um determinado momento, baseia-se na distância que o usuário se encontra do objeto.

PUC-Rio - Certificação Digital Nº 0821387/CA

Como é possível observar, existem várias otimizações que só são possíveis quando a organização hierárquica dos grafos de cena representa semântica e espacialmente a organização dos objetos na cena. Portanto, se tal disposição espacial não for considerada na modelagem do grafo de cena, essas otimizações também não são possíveis.

2.2.Grafo de Rotas

O grafo de cena modela como uma cena é estruturada (quais os seus constituintes e como eles são organizados) e como ela será renderizada (quais as suas características, as cores dos objetos, a iluminação etc.). Não está no escopo de um grafo de cena definir o comportamento da cena (como ela evolui no tempo, o que acontece, por exemplo, quando o usuário interage etc.). Geralmente, tal comportamento é definido ou por linguagens imperativas ou pelos *grafos de rotas,* que é uma abordagem declarativa.

Embora os nós no grafo de rotas sejam os mesmos nós do grafo de cena, ele é uma estrutura independente. As arestas do grafo de rotas não representam relações hierárquicas entre os nós, mas sim, rotas que interligam campos de dados dos nós. Basicamente, quando um evento altera o valor de uma propriedade de um nó, e existe uma rota entre esse campo e outro, esse novo valor é encaminhado por essa rota, gerando um "evento de saída" no nó inicial. O nó que recebe esse evento, recebendo-o como um "evento de entrada", pode alterar seu estado o que, inclusive, pode resultar na geração de novos "eventos de saída", em cascata, desde que existam outras rotas. É interessante perceber que existe uma semelhança entre as arestas nos grafos de rotas e as *sentenças causais*. Ambas informam que, quando um determinado evento ocorrer (uma condição), outro evento deve ser disparado (uma ação deve ser tomada). Contudo, as rotas sempre relacionam campos de nós que são do mesmo tipo de dados.

Geralmente, um evento de entrada "roteado" pelo grafo de rotas está relacionado com um nó de *Sensor*. Um Sensor é um nó especial responsável por gerar eventos dinamicamente. Por exemplo, existem nós de sensores *Timers,* responsáveis por gerar pulsos de relógio, nós relacionados à interação do usuário, tais como sensores de toque (ou seleção), colisão, visibilidade, proximidade, entre outros.

PUC-Rio - Certificação Digital Nº 0821387/CA

A Figura 3, a seguir, exemplifica um grafo de rotas. Esse exemplo informa que, quando um determinado sensor de toque (*TouchSensor*) é ativado, – isso acontece quando o usuário seleciona a geometria pai desse sensor – é disparado um sensor de tempo (*Timer*), que continuamente gera pulsos de relógio. Sempre que um pulso de relógio acontecer, a intensidade da luz definida pelo nó de iluminação (*PointLight*) é alterada para o valor da fração do tempo que foi alterada. Os campos de dados *touchtime* e *startTime* são do mesmo tipo de dados, assim como *fraction_changed* e *intensity*.

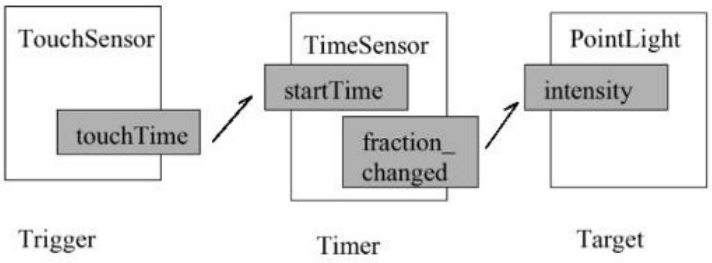

Figura 3 Exemplo de um grafo de rotas.

Como comentado anteriormente, também é possível controlar o comportamento de um grafo de cena a partir de código imperativo. A vantagem dessa última abordagem é que ela é mais expressiva quando comparada a uma abordagem declarativa (grafo de rotas, por exemplo). Por outro lado, a desvantagem está no fato de que ela restringe os autores àqueles que são programadores e conhecem uma API imperativa específica. São exemplos de APIs que permitem o controle de comportamento de grafos de cena por linguagens de programação imperativa: OpenSceneGraph (OSG COMMUNITY, 2010), Java 3D (SELMAN, 2002) e SAI (*Scene Access Interface*) (WEB3D CONSORTIUM, 2009b) – parte da especificação de X3D (WEB3D CONSORTIUM, 2009a).

Como será apresentado no Capítulo 4, a proposta deste trabalho é, assim como na abordagem por grafos de rotas, permitir ao autor especificar o comportamento da cena a partir de uma linguagem declarativa. Contudo, a vantagem residirá no fato de que o modelo utilizado, o NCM, é mais expressivo do que o grafo de rotas, como é possível observar na próxima seção.

PUC-Rio - Certificação Digital Nº 0821387/CA

2.3.NCL

Como introduzido no Capítulo 1, *Nested Context Language* (NCL) é a linguagem declarativa padrão para o sistema de TV Digital Terrestre - ISDB-T_B, e Recomendação ITU-T para serviços IPTV. *Nested Context Model* (NCM) é o modelo conceitual em que se apóia a especificação da linguagem NCL. Esta seção apresenta o NCM e a NCL, com foco nos conceitos que serão úteis no decorrer desta dissertação. Em especial, os conceitos de objetos de mídia, âncoras de propriedade, âncoras de conteúdo, elos e conectores serão de grande importância para o Capítulo 4.

Dando continuidade a esta seção, a Subseção 2.3.1 apresenta o NCM e a Subseção 2.3.2 demonstra como os conceitos do NCM são mapeados para uma aplicação XML, a linguagem NCL.

2.3.1.Nested Context Model

NCM é um modelo conceitual com foco na representação e manipulação de documentos hipermídia (SOARES e RODRIGUES, 2005). Como apresentado na Introdução, o NCM é baseado nos conceitos usuais de *nós* e *elos* hipermídia. Um *nó* representa um fragmento de informação, enquanto os *elos* relacionam esses nós. Tanto nós como elos são vistos como entidades de primeira classe do modelo. Todas as entidades no NCM possuem uma lista de propriedades, sendo diferenciadas justamente pelas propriedades que possuem.

É possível também definir *âncoras* em nós. Uma *âncora de conteúdo*, por exemplo, representa uma porção do conteúdo de um nó (uma região de uma imagem, um intervalo de tempo em um vídeo etc.). *Âncoras de propriedade*, por outro lado, representam atributos desses nós, por exemplo, largura, altura, transparência etc.

Os nós podem ser divididos em dois tipos principais: nós de conteúdo e nós de composição. Os *nós de conteúdo* (ou nós de mídia) representam objetos de mídia usuais, tais como áudio, vídeo, texto etc. Um *nó de composição* (ou, simplesmente, composição) é um nó cujo conteúdo é uma coleção de outros nós (nós de conteúdo ou nós de composição). Os nós de composição têm o objetivo de agrupar e encapsular entidades que fazem parte do seu conteúdo. Para que nós internos a uma composição possam ser vistos fora desta composição, devem-se

PUC-Rio - Certificação Digital Nº 0821387/CA

criar interfaces para esses nós, as *portas*. Os nós de composição são, por sua vez, especializados em duas classes: nós de contextos e nós de alternativa.

Os *nós de contexto* contêm um conjunto de nós de conteúdo, de contexto ou de alternativa. Adicionalmente, os nós de contexto também possuem um conjunto de elos que relacionam os nós que fazem parte do contexto. Os nós de contexto podem ser utilizados, por exemplo, para organizar lógica e hierarquicamente documentos hipermídia, favorecendo também o reúso.

Nós de alternativa, por outro lado, dão suporte à adaptação de conteúdo, agrupando nós e permitindo que apenas um deles seja selecionado para execução a partir da avaliação de regras. As *regras* são expressões que avaliam o valor de informações globais (localização do usuário, idade, informações da plataforma, criadas pelo próprio autor etc.) baseado em um operador de comparação (“=”, “<”, “>”, “<=”, “>=”, entre outros). Existem também as *regras compostas* que definem expressões lógicas e permitem unir duas ou mais regras simples por meio dos operadores lógicos “E” e “OU”.

O suporte a sincronismo do NCM é baseado nas *máquinas de estados dos eventos*. Um evento é uma ocorrência no tempo, instantânea ou com uma duração. Um *evento de apresentação*, por exemplo, representa a exibição de uma âncora de conteúdo; um *evento de seleção* representa a seleção, por parte do usuário, de uma âncora de conteúdo; e um *evento de atribuição* refere-se à alteração de uma âncora de propriedade de um nó de conteúdo ou composição. A Figura 4 apresenta os estados possíveis de um evento.

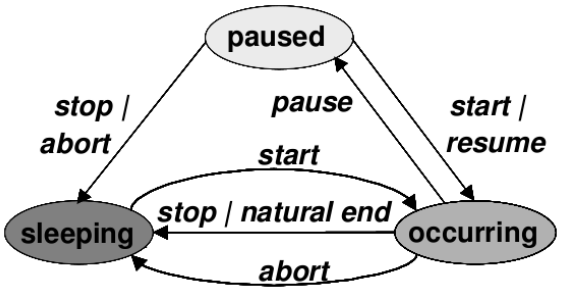

Figura 4 Máquina de estados de eventos NCM. Fonte: (SOARES e RODRIGUES, 2005)

Os elos NCM representam *relacionamentos* entre máquinas de estados de eventos associados aos nós e são definidos com base em relações. O *conector* ou

PUC-Rio - Certificação Digital Nº 0821387/CA

relação[1], também uma entidade de primeira ordem do modelo, permite especificar relações hipermídia. O modelo possibilita a definição de novos tipos de eventos e novas relações, aumentando assim, a sua expressividade. Essa é uma das principais diferenças entre o NCM e outros modelos hipermídia, visto que a grande maioria desses últimos não permite que o autor defina novas relações hipermídia. Ao criar um conector, o autor define qual a semântica da relação que está criando. Se for um conector causal, por exemplo, o autor deve informar quais as condições necessárias para que um elo que utilize esse conector seja ativado e quais as ações resultantes. Para cada condição e ação, é definido um papel que deve ser mapeado para âncoras de nós no momento da criação do elo.

Os nós de mídia NCM representam **quais** objetos de mídia serão executados (apresentados) e **como**, por meio de suas propriedades. Entidades *descritores* definem o valor inicial das propriedades relacionadas à apresentação de um nó de mídia. Os descritores podem definir parâmetros como transparência, posição, altura, volume do áudio etc. Como o descritor também é uma entidade de primeira ordem do modelo, o mesmo descritor pode ser reutilizado por vários objetos de mídias.

Adicionalmente, também é possível fazer adaptação na forma como os objetos são apresentados, por meio de *descritores alternativos*. Assim como para os nós de alternativa, a escolha de qual descritor é utilizado por um objeto de mídia, em um determinado momento, também pode ser baseada na avaliação de regras.

A Figura 5 mostra um exemplo de um documento hipermídia composto por nós, elos e nós de composição.

PUC-Rio - Certificação Digital Nº 0821387/CA

[1] A diferença entre relação e relacionamento pode ser mais bem compreendida por meio de um exemplo. Quando dizemos: "João é casado com Maria.", estamos nos referindo a um relacionamento. Por outro lado, quando falamos em "casamento", falamos de uma relação. Observe que nesse exemplo, a relação (casamento) só admite que uma pessoa faça o papel de marido e outra pessoa o de esposa. Sendo assim, observa-se que a relação, além de definir os papéis (marido e esposa) que podem ser exercidos em relacionamentos também pode definir a cardinalidade de cada um desses papéis.

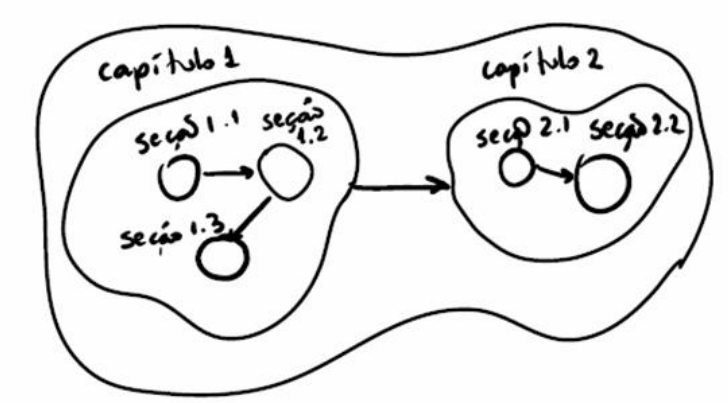

Figura 5 Nós, elos e nós de composição. Fonte: (SOARES NETO, SOARES, *et al.*, 2010)

2.3.2.Nested Context Language

A linguagem NCL é uma aplicação XML baseada no NCM. Ela mapeia as entidades e propriedades do NCM em elementos e atributos XML, com as devidas adaptações. No perfil atual para TV digital, NCL ainda não traz todas as funcionalidades do modelo NCM. Relações de restrição, por exemplo, ainda não são suportadas por esse perfil, embora estejam previstas no modelo. Além disso, as relações causais são limitadas a eventos de apresentação, seleção e atribuição, tanto no perfil para TV Digital como no perfil *Full Language*.

Um documento NCL apenas define como os objetos de mídia são estruturados e relacionados no tempo e espaço. Como uma linguagem de cola, NCL não restringe ou prescreve os tipos de conteúdo dos objetos de mídia. Nesse sentido, podem-se ter objetos de imagem (GIF, JPEG, ...), de vídeo (MPEG, MOV, ...), de áudio (MP3, WMA, ...), de texto (TXT, PDF, ...), imperativos (Xlet, NCLua, ...) e, até mesmo, outros objetos de mídia com conteúdo declarativo (outras aplicações NCL embutidas, SMIL, SVG etc.). Quais objetos de mídia são suportados depende dos exibidores de mídia que estão acoplados ao formatador NCL (exibidor NCL) (ABNT, 2007).

Ao criar uma apresentação multimídia em NCL, o autor deve informar ao formatador NCL **o que** ele deverá apresentar, **como**, **onde** e **quando**. Um documento NCL é um arquivo XML dividido em duas partes principais: o cabeçalho (elemento <head>) e o corpo (elemento <body>). No cabeçalho, estão as bases de informação que devem especificar **onde** e **como** o conteúdo deve ser

PUC-Rio - Certificação Digital Nº 0821387/CA

exibido. No corpo está descrito **o que** é o conteúdo a ser exibido e **quando** isso deve ser feito.

Por meio dos objetos de mídia (representados pelos elementos <media>), define-se **o que** apresentar, ou seja, qual conteúdo é exibido. Como já mencionado, uma mídia pode ser um áudio, vídeo, texto, imagem e, até mesmo, códigos imperativos e declarativos. Nós de mídia são definidos dentro de nós de composição, por exemplo, nós de contexto (representados pelos elementos <context> e <body>) ou de alternativa (representados pelos elementos <switch>).

Para que seja possível acessar um nó interno a um contexto, é necessário definir nesse contexto um ponto de entrada ou porta (elemento <port>) que aponte para esse nó interno. O corpo do documento (elemento <body>) também tem a semântica de um contexto, sendo tratado pelo exibidor como o contexto principal do documento. Ao encapsular outros nós, um nó de composição também tem as funções de estruturar e permitir o reúso de partes de um documento NCL.

O conjunto de atributos que define **onde** um objeto de mídia será apresentado é chamado de região[2] em NCL. Contudo, os nós de mídia não necessitam carregar em sua especificação a informação de **onde** eles devem ser exibidos. Isso pode, alternativamente, ser definido em elementos de primeira classe (elementos <region>). Um elemento <region> define uma área da tela onde uma mídia pode ser apresentada. Mais uma vez, a separação entre o conteúdo do objeto de mídia e onde ele será apresentado favorece o reúso na linguagem. Atualmente, as regiões de NCL são apenas regiões bidimensionais. As regiões são definidas por meio de sua posição (*top*, *bottom*, *left* e *right*), sua altura (*height*) ou sua largura (*width*). Adicionalmente, todo elemento <region> deve possuir um identificador único (atributo *id*), que é utilizado pelo descritor para associá-la a um ou mais objetos de mídia.

Os descritores (elementos <descriptor>) definem **como** os nós de mídia são *inicialmente* apresentados. O elemento <descriptor> possui um atributo *region* que deve referenciar um identificador de um elemento <region>. O descritor também possui um atributo *id* que é único e utilizado pelo objeto de mídia, para se associar a esse descritor. Além disso, o descritor também pode definir outros

[2] Uma discussão detalhada sobre o conceito de região em NCL e como ela pode ser especificada na linguagem – no próprio objeto de mídia, no elemento <region> ou em descritores – é apresentada no Capítulo 5.

PUC-Rio - Certificação Digital Nº 0821387/CA

atributos relacionados à como o objeto de mídia será apresentado, tais como: transparência, volume do áudio, duração explícita do objeto etc.

Uma vez informados o que, onde e como, resta então especificar **quando** os objetos de mídia serão apresentados. A ação de executar um documento NCL em um exibidor é a de apresentar o nó de contexto representado pelo corpo do documento (<body>). Apresentar um nó de contexto é equivalente a apresentar os elementos apontados por todas as portas desse contexto, em paralelo. Com isso, já é possível informar ao exibidor quais as mídias que serão executadas ao ser iniciada a apresentação do documento. A Figura 6 mostra um exemplo de código NCL que apresenta um vídeo que ocupa toda a tela.

```
1:  <?xml version="1.0" encoding="UTF-8"?>
2:  <ncl id="exemplo01"
    xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
3:    <head>
4:      <regionBase>
5:        <region id="rgVideo" height="100%" width="100%"/>
6:      </regionBase>
7:      <descriptorBase>
8:        <descriptor id="dsVideo" region="rgVideo"/>
9:      </descriptorBase>
10:   </head>
11:   <body>
12:     <port id="porta" component="video01"/>
13:     <media id="video01" src="media/video01.mp4"
    descriptor="dsVideo"/>
14:   </body>
15: </ncl>
```

Figura 6 Exemplo de código NCL que apresenta um vídeo em tela cheia.

Outra forma de especificar quando os objetos de mídia serão exibidos é por meio de elos. Os elos (elementos <link>) permitem que relacionamentos de sincronismo mais complexos sejam suportados por NCL. Para facilitar a definição dos elos, eles são escritos tendo como base relações hipermídia (em NCM, denominadas *conectores*). NCL, em seu perfil *Full Language* possui tanto relações causais como relações de restrição. Em seu perfil para TV digital, entretanto, só são suportadas relações causais. Essas relações são especificadas utilizando conectores.

Os conectores são definidos no elemento <connectorBase> dentro do cabeçalho (elemento <head>). Embora os conectores sejam entidades de primeira classe, é interessante que eles sejam definidos por usuários experientes com a linguagem e então disponibilizados para aqueles usuários com menos experiência apenas os reutilizarem. Para importar bases de conectores em um documento, é

PUC-Rio - Certificação Digital Nº 0821387/CA

possível definir elementos <importBase> como filhos do elemento <connectorBase>.

Um conector causal (<causalConnector>) é definido como um conjunto de papéis de condição e um conjunto de papéis de ação e tem um identificador único (atributo *id*), que será utilizado pelo elo. Caso o conjunto de condições seja satisfeita, as ações resultantes serão executadas. Em um conector é necessário que se defina pelo menos uma condição e uma ação. Cada condição ou ação está associada a um papel (*role*) que, posteriormente, deve ser associado à interface (âncora ou porta) de um nó, através dos elos, pelos elementos <bind>. As condições e ações em NCL, hoje, podem estar associadas à apresentação, seleção ou atribuição. O código NCL da Figura 7 mostra um exemplo de um elo informando que "quando o nó de mídia *video1* terminar sua exibição, o nó de mídia *nolua* deve iniciar sua exibição".

```
1: <?xml version="1.0" encoding="UTF-8"?>
2: <ncl id="exemplo02"
   xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
3:   <head>
4:     <!-- ... -->
5:     <connectorBase>
6:       <causalConnector id="onEndStart">
7:         <simpleCondition role="onEnd"/>
8:         <simpleAction role="start"/>
9:       </causalConnector>
10:    </connectorBase>
11:    <!-- ...-->
12:  </head>
13:  <body>
14:    <!-- ... -->
15:    <link xconnector="onEndStart">
16:      <bind component="video1" role="onEnd"/>
17:      <bind component="nolua" role="start"/>
18:    </link>
19:    <!-- ... -->
20:  </body>
21:</ncl>
```

Figura 7 Exemplo de elo (relacionamento) e conector (relação) em NCL.

Sincronismo temporal e espacial, adaptação de conteúdo e exibição de mídias em múltiplos dispositivos de exibição são as principais funcionalidades que o NCM e a NCL permitem expressar. Dessa forma, percebe-se que o NCM é um modelo bem mais robusto do que o grafo de rotas, que se restringe a sentenças causais. As sentenças causais nos grafos de rotas, ainda assim, não permitem definir condições nem ações compostas, sendo, por isso, menos expressivas do que as sentenças causais NCL. Justamente por isso, uma das propostas deste

PUC-Rio - Certificação Digital Nº 0821387/CA

trabalho, discutida no Capítulo 4, é embutir em documentos NCL objetos 3D representados por grafos de cena. Como uma extensão natural aos mecanismos de âncoras de NCL, tornar-se-á possível controlar o comportamento desses objetos representados por grafos de rotas por meio de elos e conectores NCM.

PUC-Rio - Certificação Digital Nº 0821387/CA