Capítulo 1

Desafios da Verificação de Consistência de Documentos Multimídia Interativos

Joel dos Santos[1 2] e Débora Muchaluat-Saade[2]

[1] Escola de Informática & Computação - CEFET/RJ
[2] Laboratório MídiaCom - Dep. de Ciência da Computação - Universidade Federal Fluminense

Abstract

A multimedia document may be specified manually or automatically generated, instantiated, adapted or has its content and structure dynamically edited. Therefore, changes in a document specification may occur from its creation *to its* execution*. Such different phases and changes performed over a document caracterize its life cycle. It is important to maintain multimedia document consistency along its life cycle, which means that document execution should always follow guidelines expressed at* creation *time. Different works in the literature present approaches for addressing this issue by validating multimedia documents along different phases of a document life cycle. This work discusses challenges and approaches for multimedia document consistency checking along its life cycle. There will be a discussion about key solutions presented in the literature, challenges and directions for future research in this topic.*

Resumo

Um documento multimídia pode ser especificado manualmente ou gerado automaticamente, instanciado, adaptado ou ter seu conteúdo e estrutura dinamicamente editada. Portanto, mudanças na especificação de um documento podem ocorrer desde sua criação *até sua* execução*. Estas diferentes fases e mudanças realizadas em um documento caracterizam o seu ciclo de vida. É importante manter a consistência de um documento multimídia ao longo do seu ciclo de vida, o que significa que a execução de um documento deveria sempre seguir diretrizes expressas em tempo de* criação*. Diferentes trabalhos na literatura apresentam abordagens para resolver esse problema através da validação de documentos multimídia em diferentes fases ao longo do seu ciclo de vida. Este capítulo*

tem como principal objetivo discutir os desafios e abordagens para a verificação de consistência de documentos multimídia ao longo de seu ciclo de vida. Serão discutidas as principais soluções apresentadas na literatura e apresentados os desafios e pontos em aberto que podem nortear pesquisas futuras.

1.1. Introdução

Atualmente, observamos uma grande difusão de diferentes dispositivos portáteis equipados com câmeras, microfones, sensores, displays e com grande capacidade de processamento e armazenamento. Com isso, sistemas multimídia tornam-se cada vez mais presentes em nosso dia-a-dia. Podemos ser tanto produtores de conteúdo multimídia, dada a capacidade de captura de informação de tais dispositivos, quanto consumidores, dada a capacidade de exibição dos mesmos.

Tais conteúdos representam unidades de informação consumidas por seres humanos, como um texto, uma imagem, um áudio, um vídeo e mesmo efeitos sensoriais, como movimento ou vibração. Estas informações, denominadas mídias, são apresentadas de forma organizada no tempo e espaço segundo a descrição contida em um documento multimídia. Um documento multimídia, portanto, descreve o conjunto de mídias que serão por ele consideradas - também chamado de o conteúdo de um documento - e relações entre elas para determinar como tais mídias serão apresentadas no tempo, espaço ou ambos. Relações em um documento podem levar em consideração ocorrências de eventos, como interação do espectador (incluindo reconhecimento de gestos), o resultado de uma computação (por exemplo por um script auxiliar) ou uma *query* (por exemplo a um servidor externo) realizada durante a execução do documento ou mesmo a chegada do espectador a alguma posição geográfica.

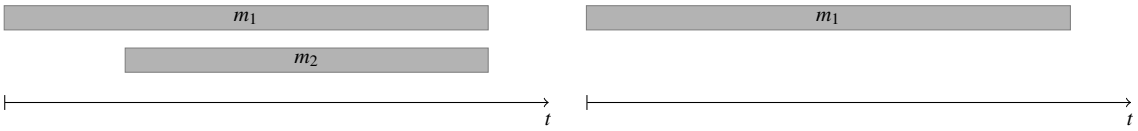
Quando executada, a descrição contida em um documento resulta em um arranjo particular das mídias no tempo e espaço, chamado de uma apresentação multimídia. Diferente de um documento, uma apresentação é o resultado que é efetivamente apresentado ao espectador.

Atualmente, novas facilidades estão disponíveis para a apresentação de documentos multimídia, dentre as quais podemos destacar: a personalização de uma apresentação para um dado espectador [Laborie et al. 2011]; a divisão de uma apresentação em múltiplos dispositivos (por exemplo um aparelho de TV e uma segunda tela) [Sarkis et al. 2014]; novos dispositivos de exibição (como telas *touch* ou com sensores de movimento embutidos) melhorando a interface da apresentação com o espectador e permitindo novas formas de interação (como gestos, *multi touch* e *force touch*); edição dinâmica de uma apresentação (por exemplo através de anotações [Teixeira et al. 2012] ou edição ao vivo [Soares et al. 2012]); e a geração dinâmica do conteúdo de documentos multimídia de acordo com uma busca feita pelo espectador (caso frequente em documentos web).

Tendo em vista tais facilidades, um documento multimídia descreve um conjunto de possíveis diferentes apresentações. Este conjunto é restringido de acordo com o contexto do espectador de forma a escolher a apresentação mais adequada a ele. O contexto do espectador contém informação a respeito do seu dispositivo de exibição, preferências de exibição, e tudo mais que possa influenciar na apresentação do documento. Apesar

de um documento resultar em diferentes apresentações, elas não são completamente diferentes umas das outras, visto que devem seguir uma mesma especificação contida em um documento. Exemplos de adaptação de um documento ao contexto do espectador são ajustar uma apresentação ao tamanho da tela do dispositivo de exibição, a supressão de um áudio da apresentação dada a indisponibilidade de um canal de áudio, etc.

Após um documento ser moldado de acordo com o contexto do espectador, este pode ser ainda modificado de acordo com a interação do espectador, resultando assim na apresentação final, isto é, aquela efetivamente apresentada ao espectador. A interação do espectador pode acontecer, por exemplo, via gestos, reconhecimento de fala, seleção, *multi touch* e *force touch*. Como exemplo, considere um documento onde uma mídia m_2 é apresentada dado que o espectador tenha interagido com a mídia m_1. A Figura 1.1 apresenta duas possíveis apresentações obtidas a partir de tal documento, ambas diferindo pela ocorrência ou não de interação do espectador.



(a) Espectador interagiu com m_1 (b) Espectador não interagiu com m_1

Figura 1.1: Possíveis apresentações de acordo com interação do espectador

A descrição contida em um documento multimídia é usualmente textual, seguindo alguma linguagem de autoria. No paradigma declarativo, linguagens de autoria provêem construções com um alto grau de abstração para declarar o conjunto de mídias em um documento e as relações entre elas. A ideia de uma linguagem declarativa é separar a descrição contida em um documento das especificidades de sua execução. O nível de abstração das construções providas por uma linguagem varia de acordo com o modelo de sincronização por ela seguido. Na Seção 1.2 serão apresentados diferentes modelos de sincronização encontrados na literatura.

Um outro benefício da autoria declarativa, no cenário multimídia, é facilitar a tarefa de criação de documentos multimídia. Tal objetivo é importante visto que documentos multimídia podem ser utilizados em diferentes áreas, como web, TV digital e IPTV; e podem ser criados por autores com diferentes perfis, como desenvolvedores e produtores de conteúdo.

A especificação contida em um documento pode variar ao longo de seu ciclo de vida, isto é, desde sua criação até sua execução. Ela varia de relações de sincronização expressas em uma linguagem de autoria em alto nível de abstração até eventos de execução de baixo nível e modificações incrementais que podem ser feitas sobre um documento. Portanto, é importante garantir que um documento permaneça consistente à sua especificação ao longo do seu ciclo de vida, isto é, às diretrizes expressas durante sua criação.

A consistência de um documento ao longo do seu ciclo de vida pode ser mantida através de validação. A ideia é combinar a especificação de um documento com propriedades representando as diretrizes e validá-las de forma a garantir que qualquer modificação produzida em um dado passo do ciclo de vida não torne o documento inconsistente.

Este capítulo tem por objetivo discutir a verificação de consistência de documentos multimídia declarativos ao longo do seu ciclo de vida, seus desafios e diferentes abordagens. Serão apresentadas as principais soluções encontradas na literatura e apresentados os desafios e pontos em aberto que podem nortear pesquisas futuras.

O restante deste capítulo está estruturado da seguinte forma.

A Seção 1.2 apresenta os diferentes modelos de sincronização espaço-temporal utilizados na autoria de documentos multimídia. Cada modelo define uma abordagem para a especificação da sincronização entre objetos de mídia ao longo da execução de um documento multimídia. Tais abordagens serão usadas para a classificação das soluções sobre verificação de consistência de documentos apresentadas na literatura.

A Seção 1.3 discute as diferentes formas de realizar a validação de documentos multimídia, dentre as quais destacam-se a validação estrutural e comportamental. A primeira visa validar a forma como um documento é descrito e a segunda visa validar o comportamento de um documento quando executado. A validação comportamental, por sua vez, pode ser dividida em três categorias, dependendo do tipo de validação realizada. São elas: a validação temporal, a validação espacial e a validação espaço-temporal.

A Seção 1.4 apresenta as diferentes etapas pelas quais passa um documento, juntamente com a classificação de tais etapas em um ciclo de vida. A apresentação do ciclo de vida é feita juntamente com a discussão das etapas onde a validação de um documento é importante e a apresentação de como padrões e trabalhos publicados na literatura se encaixam nesse ciclo.

A Seção 1.5 apresenta trabalhos publicados na literatura relacionados com a validação de documentos multimídia, juntamente com a indicação da etapa do ciclo de vida onde atuam. Ainda, as soluções apresentadas são classificadas de acordo com o tipo de validação que provêem.

A Seção 1.6 apresenta um exemplo de documento multimídia juntamente com a verificação de sua consistência.

A Seção 1.7 discute os desafios de pesquisa relacionados à validação de documentos multimídia. Esses desafios são problemas em aberto, ainda não tratados pelos trabalhos relacionados e que podem motivar novas pesquisas na área.

Por fim, a Seção 1.8 conclui este capítulo.

1.2. Modelos de Sincronização Temporal

Um modelo de sincronização temporal especifica como são definidas as relações espaço-temporais entre as mídias que compõem um documento. Construções comuns em linguagens declarativas são: nós, fragmentos de nós (âncoras ou simplesmente fragmentos), elos e composições. Nós são entidades que representam mídias em um documento, sendo a representação do seu conteúdo abstraída em sua representação no documento. Um nó pode representar um texto, áudio, vídeo, imagem, script, um programa, etc. O conteúdo de um nó é composto de um conjunto de unidades de informação dependentes do tipo de mídia (quadros de um vídeo, amostras de um áudio, pixels em uma imagem, caracteres em um texto, etc). Um fragmento de um nó representa uma parte, um subconjunto, de seu

conteúdo. Elos, por sua vez, são usados para representar relações entre nós. Finalmente, composições, por sua vez, representam conjuntos de nós e/ou elos ou outras composições, e também são usadas para representar relações entre nós.

Nesta seção, os conceitos acima serão utilizados na descrição das diferentes classes de modelos de sincronização temporal. As seguintes classes serão apresentadas: baseado em eixos temporais, baseado em restrições, baseado em sincronização hierárquica, baseado em especificação formal, baseado em eventos e baseado em scripts.

Para ilustrar os diferentes modelos de sincronização temporal, a Figura 1.2 apresenta um exemplo de organização temporal de um conjunto de mídias. Neste exemplo temos um vídeo, um áudio, uma imagem e um texto relacionados no tempo. Este exemplo foi retirado de [Boll 2001].

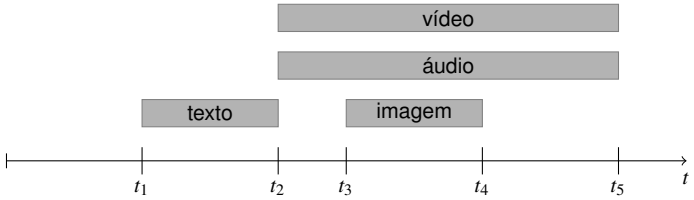


Figura 1.2: Exemplo de cenário temporal

A Figura 1.2 descreve a apresentação de um texto, seguido pela apresentação, em paralelo, de um vídeo e um áudio. Com um atraso de 1 minuto ($t_3 = t_2 + 1$) após o término do texto, uma imagem é apresentada. As seções a seguir apresentam as diferentes classes de modelos de sincronização temporal e como cada modelo representa o exemplo descrito.

1.2.1. Baseado em Eixos Temporais

Modelos de sincronização baseados em eixos temporais, ou baseado em linha de tempo, mapeiam o início ou o fim da apresentação de um nó em um eixo temporal. Este eixo pode ser global, de forma que todo nó é mapeado no mesmo eixo temporal, ou pode ser virtual, de forma que nós podem ser mapeados em diferentes eixos. Exemplos de sistemas que usam este tipo de sincronização temporal são softwares comerciais como Apple iMovie ou Apple Final Cut. A representação do exemplo da Figura 1.2 usando um modelo de sincronização baseado em eixo temporal é apresentada a seguir.

```
1  texto [t1, t2]
2  vídeo [t2, t5]
3  áudio [t2, t5]
4  imagem [t3, t4]
```

1.2.2. Baseado em Restrições

Modelos de sincronização baseados em restrições, definem restrições sobre a ordem temporal de um conjunto de nós. Modelos de sincronização baseados em restrições podem ser divididos em dois tipos: baseados em intervalos e em pontos de referência.

A *sincronização baseada em intervalos* representa a duração de um nó como um intervalo. A sincronização entre dois intervalos é definida usando treze relações básicas

apresentadas por Allen em [Allen 1983], ou sete se as relações inversas forem desconsideradas. Uma outra abordagem é utilizar as vinte e nove relações definidas por Wahl em [Wahl and Rothermel 1994]. A Figura 1.3 apresenta as sete relações de Allen.

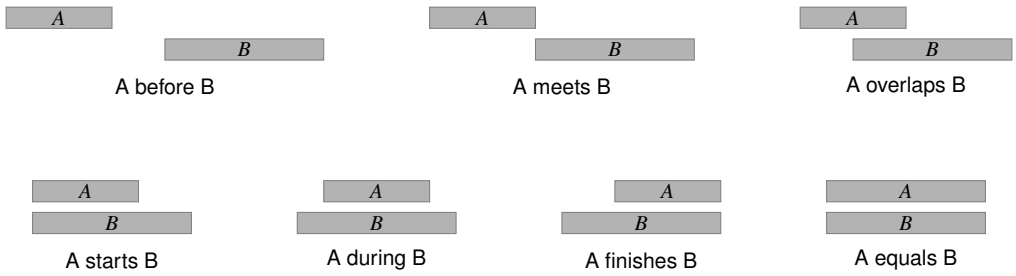


Figura 1.3: Relações de Allen entre intervalos temporais

Um exemplo de sistema que usa este tipo de sincronização temporal é o sistema de autoria Madeus [Jourdan et al. 1998]. A representação do exemplo na Figura 1.2 usando um modelo de sincronização baseado em intervalos é apresentada a seguir, onde assume-se que as mídias texto e imagem possuem duração associada.

```
1  vídeo equals áudio
2  texto meets vídeo
3  texto meets imagem delay 1min
```

Note que, apesar de esse tipo de sincronização prover uma definição simples das relações entre nós (intervalos), um modelo de sincronização baseado em intervalos não permite a especificação de relacionamentos entre fragmentos de um nó. Esse tipo de relacionamento é feito de forma indireta através de atrasos ou subdividindo um nó de acordo com seus fragmentos.

A *sincronização baseada em pontos de referência* usa pontos de referência para definir a sincronização entre nós. Pontos de referência podem ser o início ou fim da apresentação de um nó ou de um de seus fragmentos. O relacionamento entre nós (ou fragmentos) é feito através da especificação de restrições sobre os pontos de referência, criando pontos de sincronização. Nós (ou fragmentos) participando de um mesmo ponto de sincronização são iniciados ou parados de uma única vez, quando o ponto de sincronização é alcançado durante a apresentação do documento. A representação do exemplo na Figura 1.2 usando um modelo de sincronização baseado em pontos de referência é apresentada a seguir. Assim como no caso anterior, assume-se que as mídias texto e imagem possuem duração associada.

```
1  Point t2:
2      texto ends and vídeo starts and áudio starts and imagem starts delay 1 min
3  Point t5:
4      vídeo ends and áudio ends
```

Dado que seja possível a representação de fragmentos de um nó, pontos de referência podem ser criados para definir sincronização para tais fragmentos. Assim, a sincronização baseada em pontos de referência apresenta maior flexibilidade que a baseada em intervalos. Um exemplo de sistema que usa sincronização baseada em pontos de referência é o sistema Firefly [Buchanan and Zellweger 1992, Buchanan and Zellweger 2005].

1.2.3. Baseado em Sincronização Hierárquica

Modelos baseados em sincronização hierárquica descrevem a sincronização entre nós usando duas composições temporais principais: composição sequencial e paralela. Nessa abordagem, a sincronização do documento é definida por uma árvore de composições temporais, representando a apresentação sequencial ou paralela do seus nós internos. É possível ainda a definição de um atraso para o início de um nó específico. Exemplos de uso de modelo baseado em sincronização hierárquica são as linguagens declarativas SMIL (Synchronized Multimedia Integration Language) [W3C 2008b] e MPEG-4 XMT (eXtensible MPEG-4 Textual) [ISO/IEC 2005]. A representação do exemplo na Figura 1.2 usando um modelo baseado em sincronização hierárquica é apresentada a seguir.

```
1  seq{
2      texto
3      par{
4          vídeo
5          áudio
6          imagem begin 1min
7      }
8  }
```

Um modelo baseado em sincronização hierárquica provê uma abordagem simples para a sincronização de nós. Entretanto, assim como na sincronização baseada em intervalos, não é possível a criação de relacionamentos entre fragmentos dos nós, sendo, este tipo de sincronização, feito de forma indireta.

1.2.4. Baseado em Especificação Formal

Modelos de sincronização baseados em especificação formal são aqueles que usam algum tipo de formalismo para representar nós e especificar a sincronização entre eles. Esse tipo de sincronização se beneficia do grande número de ferramentas para modelos formais disponíveis para apresentar o documento descrito. Dois tipos comuns de modelos de sincronização baseados em especificação formal são: sincronização baseada em redes de Petri temporizadas [Peterson 1981] e baseada em *statecharts* [Harel 1987].

A *sincronização baseada em redes de Petri temporizadas* usa redes de Petri, onde lugares possuem duração e representam os nós do documento e transições são utilizadas para a sincronização entre lugares. Apesar de permitir a definição de qualquer tipo de relação de sincronização, esse tipo de modelo também não permite a criação de relacionamentos entre fragmentos dos nós. Exemplos de sistemas que utilizam sincronização baseada em redes de Petri são os sistemas Trellis [Furuta and Stotts 2001], caT (*Context-aware Trellis*) [Na and Furuta 2001] e HTSPN (*Hierarchical Time Stream Petri Net*) [Willrich et al. 2001]. Uma representação do exemplo na Figura 1.2 usando um modelo de sincronização baseado em redes de Petri é apresentada na Figura 1.4.

A *sincronização baseada em statecharts* utiliza statecharts para representar os nós e sua sincronização temporal. Nessa abordagem estados representam mídias, enquanto transições definem as relações de sincronização entre estados. Uma transição ainda define se estados filhos são apresentados simultaneamente ou não. Um exemplo de sincronização baseada em statecharts é apresentada em [de Oliveira et al. 2001]. A representação do exemplo na Figura 1.2 usando um modelo de sincronização baseado em statecharts é apresentada na Figura 1.5.

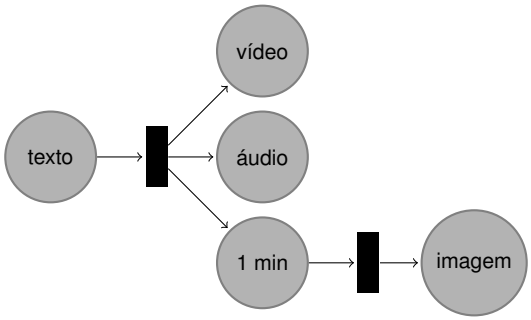


Figura 1.4: Representação baseada em Redes de Petri

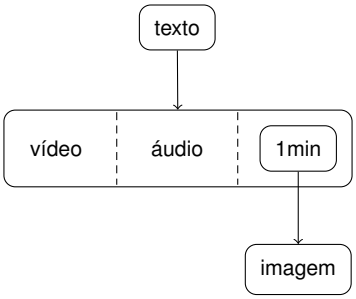


Figura 1.5: Representação baseada em Statecharts

Na Figura 1.5, os estados do *statechart* representam nós do documento. Ainda, um estado é utilizado para representar o atraso de 1 minuto para o início da imagem. Neste exemplo, a composição de estados, como visto para o vídeo, o áudio e o atraso de 1 minuto, indica que tais mídias (e atraso) iniciarão sua execução em paralelo. As transições entre estados indicam a sequência de execução das mídias representadas em cada estado.

1.2.5. Baseado em Eventos

Modelos de sincronização baseados em eventos permitem a criação de relacionamentos entre eventos que acontecem durante a execução do documento, como eventos de apresentação do conteúdo de um nó, eventos de seleção (interação do espectador, por exemplo) ou eventos de atribuição (mudança do valor de uma variável, por exemplo). Exemplos de modelos baseados em eventos são os modelos NCM (*Nested Context Model*) [Soares et al. 2000], que foi usado como base para a linguagem de autoria NCL (*Nested Context Language*) [ITU 2009], e Labyrinth [Díaz et al. 2001]. A representação do exemplo na Figura 1.2 usando um modelo de sincronização baseado em eventos é apresentada a seguir, onde assume-se que as mídias texto e imagem possuem duração associada.

```
1 onBegin document, start texto
2 onEnd texto, start vídeo and start áudio and start imagem delay 1min
3 onEnd video, end áudio
```

1.2.6. Baseado em Scripts

Modelos de sincronização baseados em scripts usam uma descrição textual para a sincronização dos nós. Um script (descrição textual) define um conjunto de passos para a

obtenção da apresentação desejada. Exemplos de uso de modelo de sincronização baseado em scripts são a linguagem HTML5 [W3C 2014], onde a sincronização entre mídias é definida através de scripts JavaScript, e a linguagem Flash [Adobe 2010]. A representação do exemplo na Figura 1.2 usando um modelo de sincronização baseado em scripts é apresentada a seguir. No exemplo, os atrasos são especificados em milissegundos.

```
1   function imagemEnd() {
2       imagem.stop();
3   }
4
5   function imagemStart() {
6       imagem.start();
7       setInterval(imagemEnd, (t4−t3)∗60000);
8   }
9
10  function vídeoEnd() {
11      vídeo.stop();
12      áudio.stop();
13  }
14
15  function textoEnd() {
16      texto.stop();
17      vídeo.start();
18      áudio.start();
19      setInterval(imagemStart, 60000);
20      setInterval(vídeoEnd, (t5−t2)∗60000);
21  }
22
23  function textoStart() {
24      texto.start();
25      setInterval(textoEnd, (t2−t1)∗60000);
26  }
27
28  textoStart();
```

Um modelo de sincronização é dito mais expressivo que um outro modelo, quando é capaz de representar apresentações mais elaboradas que o outro modelo. Scripts são muito expressivos, entretanto, uma desvantagem do uso de scripts é o fato de o autor ter de definir todos os detalhes para a sincronização dos nós através de programação, o que já é abstraído em uma abordagem declarativa.

No paradigma de programação declarativo, modelos de sincronização baseados em eventos são uma abordagem bastante expressiva, além de apresentarem um alto nível de abstração, para a autoria de documentos multimídia. Através de eventos, é possível descrever as mesmas apresentações passíveis de serem descritas com outros modelos [Blakowski and Steinmetz 1996]. Uma exceção, entretanto, é o caso de modelos baseados em restrições. Dado que, em geral, modelos de sincronização baseados em eventos definem relações causais entre eventos, a representação de restrições entre mídias, quando possível, é complexa de ser feita usando somente eventos.

Em geral, a validação de modelos de sincronização baseados em restrições é feita sobre o próprio conjunto de restrições. Por outro lado, a validação de modelos baseados em eventos (e consequentemente nos demais) é feita sobre uma representação dos estados do documento ao longo de sua execução. Tal classificação é explorada na Seção 1.5 ao apresentar as soluções disponíveis na literatura.

Nesse contexto, um modelo híbrido, isto é, suportando eventos e restrições, surge como uma solução interessante a ser explorada, possibilitando a validação de uma grande

gama de documentos multimídia declarativos.

1.3. Tipos de Validação de Documentos Multimídia

Como discutido na Seção 1.1, um documento multimídia descreve um conjunto de possíveis apresentações. Posteriormente, durante sua execução, uma única apresentação é escolhida, conforme o contexto do espectador. A apresentação resultante, portanto, representa a execução de um dado documento.

Considerando tanto sua criação quanto sua execução, um documento pode ser considerado de dois pontos de vista distintos, um representando sua especificação e outro representando sua execução. No primeiro, são consideradas as construções em uma linguagem de autoria específica utilizadas na criação de um documento. No segundo, são considerados os comportamentos resultantes de tais construções quando o documento é executado.

Da maneira análoga, inconsistências podem ser consideradas de dois pontos de vista distintos, quando certas propriedades sobre um documento não são satisfeitas. Tais propriedades são classificadas em *estruturais* e *comportamentais*. As seções a seguir discutem tais propriedades e seu uso para a validação de um documento.

1.3.1. Propriedades Estruturais

Desde sua criação, a linguagem XML (*Extensible Markup Language*) [W3C 2008a] tem sido utilizada como uma sintaxe concreta para linguagens multimídia. Elementos XML são usados para representar mídias e relações entre elas. Exemplos de linguagens declarativas baseadas em XML são HTML5 [W3C 2014], NCL [ITU 2009] e SMIL [W3C 2008b]. NCL é parte do padrão brasileiro de TV digital [ABNT 2011] e padrão ITU para serviços IPTV [ITU 2009]. SMIL é padrão para apresentações multimídia na web [W3C 2008b].

Regras sintáticas definidas pela gramática de uma linguagem de autoria multimídia induzem um conjunto de propriedades estruturais que um documento deve seguir, de forma a ser considerado estruturalmente consistente. Trabalhos publicados na literatura [Araújo et al. 2008, Neto et al. 2011] identificam as seguintes propriedades estruturais:

- A estrutura léxica e sintática de um documento deve ser bem formada e estar de acordo com a gramática da linguagem de autoria utilizada. Por exemplo, quando uma linguagem baseada em XML é utilizada, as *tags* XML [W3C 2008a] devem ser corretamente fechadas e estarem presentes no espaço de nomes (*namespace*) da linguagem em uso.
- Todo elemento em um documento deve conter somente elementos filhos válidos e com a cardinalidade correta.
- Todo elemento em um documento deve conter somente atributos válidos, sendo que os atributos obrigatórios devem estar definidos.
- Todo identificador, quando for o caso, deve ser único.
- Atributos com valores relacionados devem seguir as restrições definidas pela linguagem de autoria. Por exemplo, NCL define os atributos *type* e *subtype* para o

elemento *transition* [ITU 2009]. Para cada diferente valor do atributo *type*, NCL define um conjunto de valores que o atributo *subtype* pode assumir. Um outro exemplo em NCL são os atributos *component* e *interface*, onde um dado elemento referenciado pelo atributo *interface* deve ser um elemento filho daquele referenciado pelo atributo *component*.

- Referências entre elementos devem seguir as restrições definidas pela linguagem de autoria. Por exemplo, NCL define atributos que podem referenciar somente um tipo ou um grupo específico de elementos. No elemento *media*, o atributo *descriptor* pode referenciar apenas elementos *descriptor*, enquanto o atributo *refer* pode referenciar apenas outros elementos *media*.

- Elementos dentro de uma composição não podem referenciar elementos fora da mesma composição. Por exemplo, em SMIL a composição *par* define o atributo *endsync*, o qual pode fazer referência a um componente interno a composição, determinando que toda a composição irá terminar sua apresentação quando o componente referenciado terminar. Neste caso, o atributo *endsync* de uma composição não pode fazer referência a um componente externo a ela.

- Uma composição não pode criar um *loop* de aninhamento, isto é, uma composição não pode conter a si mesma diretamente ou através de outras composições. NCM [Soares and Rodrigues 2005] e caT [Na and Furuta 2001] permitem o reúso de composições (contextos e switches em NCM e lugares em caT). Em ambos, composições não podem aninhar a si mesmas, caso contrário o documento multimídia seria inconsistente.

- Um elemento não pode criar um *loop* de reúso, isto é, um elemento não pode reusar a si mesmo diretamente ou através de outros elementos. Por exemplo, um elemento NCL não pode fazer referência, através de seu atributo *refer* a si mesmo.

Inconsistências estruturais surgem quando um documento não segue as propriedades estruturais. A validação estrutural de um documento, portanto, é importante, visto que uma ferramenta de exibição de um documento multimídia pode não conseguir ler ou mesmo executar um documento com inconsistências estruturais.

1.3.2. Propriedades Comportamentais

Propriedades comportamentais são usadas para verificar a existência de inconsistências na apresentação resultante da execução de um documento. O conjunto de relações definido em um documento é comumente separado em dois eixos, o espacial e o temporal. Chamaremos o conjunto de relações em cada eixo como o leiaute do documento. Desta forma, dizemos que um documento define um leiaute temporal e um leiaute espacial.

No leiaute temporal de um documento, mídias são organizadas no tempo seja através de valores absolutos ou em relação a outras mídias ou ocorrências de eventos, como interação do espectador. No leiaute espacial, mídias são organizadas comumente em relação à tela do dispositivo de exibição, a outras mídias ou em canais predefinidos. É possível ainda que relações espaciais definam o posicionamento de uma mídia em relação a uma ou mais mídias declaradas no documento.

É usual que mídias sejam posicionadas relativamente à tela, em valores absolutos (*pixels*) ou relativos (porcentagem). Apesar da posição inicial das mídias ser definida de forma estática, é possível que seu posicionamento seja alterado ao longo da execução. Algumas linguagens de autoria multimídia permitem a mudança da posição de mídias em resposta à ocorrência de eventos na apresentação do documento. Tais mudanças podem considerar o movimento de mídias, por exemplo modificando seus atributos *left*/*top*, e/ou o redimensionamento de mídias, por exemplo modificando seus atributos *width*/*height*. A modificação de atributos espaciais de uma mídia pode ser instantânea ou incremental ao longo de um intervalo temporal.

É possível que o leiaute espaço-temporal resultante da execução de um documento não represente a expectativa do autor ao criar um documento. Tal quebra de expectativa pode ocorrer pelo uso indevido de construções da linguagem de autoria utilizada ou pela falta de relações no documento criado. Diretrizes do autor, portanto, representam comportamentos esperados em um documento e devem ser definidas junto da criação do documento. Tais diretrizes têm por objetivo evitar incompatibilidades entre "o que o autor quer" e "o que o autor percebe". Trabalhos publicados na literatura [Santos et al. 1998, de Oliveira et al. 2001, Júnior et al. 2012, dos Santos 2016] identificam as seguintes diretrizes gerais:

- Toda mídia de um documento deve ser apresentada durante a execução do documento.
- Uma mídia de um documento deve terminar sua apresentação.
- A execução de um documento como um todo deve terminar. A execução do documento termina se a apresentação de todos as mídias do documento terminam e não há *loops* de execução (por exemplo, uma mídia reiniciando sua apresentação toda vez que termina).
- Duas mídias distintas não devem usar um mesmo recurso de apresentação (posição na tela ou um canal de áudio, por exemplo) simultaneamente, evitando sua sobreposição.

Apesar de diferentes trabalhos publicados na literatura identificarem tais comportamentos como esperados [Santos et al. 1998, de Oliveira et al. 2001, Júnior et al. 2012, dos Santos 2016], eles podem não condizer com a intenção do autor. Por exemplo, se um documento representa um jogo, o autor pode querer que este nunca termine sua execução. Entretanto, se pensarmos em um documento representando um comercial interativo, este deverá terminar obedecendo o tempo limite a ele alocado pela emissora.

As diretrizes acima citadas representam diretrizes gerais a serem aplicadas a qualquer documento. Ainda, o autor de um documento pode definir diretrizes específicas para o documento sendo criado. Por exemplo, ele(a) pode definir que duas mídias A e B devem ser apresentadas uma após a outra durante a apresentação do documento. Tais diretrizes são identificadas como diretrizes definidas pelo autor [dos Santos 2012, dos Santos et al. 2013, dos Santos 2016].

Durante a criação de um documento, usualmente o autor de um documento tenta verificar se um documento segue suas diretrizes através de simulação e/ou execução do documento. Neste processo, o autor executa um documento diversas vezes, assumindo o papel do espectador, e observa o resultado obtido em cada execução, verificando se o mesmo se adequa ou não às suas expectativas. Esse processo, entretanto, usualmente não é *efetivo*, já que diversas execuções seriam necessárias para a verificação de comportamentos indesejáveis, e pode ser *incompleta*, já que um documento pode possuir infinitas diferentes execuções. Ainda, no caso onde um documento é gerado automaticamente dentro de uma cadeia de produção, a simulação de um documento, caso possível, pode ser muito complexa de ser realizada.

1.3.3. Validação Estrutural e Comportamental

Inconsistências estruturais surgem quando um documento não segue as propriedades apresentadas na Seção 1.3.1. Por outro lado, inconsistências comportamentais surgem quando a combinação das relações declaradas em um documento e/ou diretrizes do autor é inconsistente, como apresentado na Seção 1.3.2.

A validação estrutural de um documento multimídia é importante uma vez que um documento estruturalmente inconsistente pode não conseguir ser executado ou mesmo lido. Ainda, a validação estrutural deve ser feita antes da validação comportamental, uma vez que inconsistências estruturais podem impossibilitar a validação comportamental do documento.

O tipo de validação comportamental provido depende do tipo de relações e propriedades suportadas pela ferramenta de validação. Nesta seção, para ilustrar os diferentes tipos de validação comportamental, serão apresentados dois pequenos exemplos de um documento d que apresenta duas mídias A e B. Para cada exemplo, as posições de A e B são definidas em valores absolutos. Cada figura representa um exemplo, onde retângulos tracejados representam a tela do dispositivo de exibição em um dado momento e retângulos sólidos representam a região da tela onde uma mídia é apresentada. Setas entre duas telas representam um salto no tempo e setas entre regiões representam um movimento no espaço. No caso de o movimento ser contínuo em um intervalo, a duração do intervalo é indicada junto a seta.

1.3.3.1. Caso 1

O exemplo do Caso 1 descreve um leiaute espacial estático, isto é, A e B não mudam sua posição ao longo do tempo. Este caso é um exemplo puramente temporal, onde A é apresentada (imediatamente) antes de B no tempo. A Figura 1.6 apresenta o leiaute espaço-temporal percebido pelo espectador do documento d.

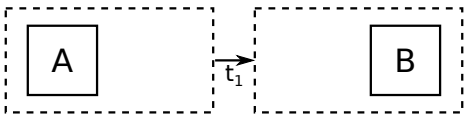


Figura 1.6: Leiaute espaço-temporal do caso 1

Considerando o exemplo acima, a maior parte das diretrizes do autor podem ser descritas através de propriedades temporais. Por exemplo, o autor pode querer garantir que, para este documento, A seja apresentado antes de B, o que pode ser representado pela fórmula A *before* B. Ainda, o autor pode querer garantir que as mídias A e B não sejam apresentadas ao mesmo tempo, o que pode ser representado pela fórmula $not(A$ *together* $B)$.

Apesar de ser um exemplo puramente temporal, o autor pode ainda expressar propriedades espaciais. Por exemplo, o autor pode querer garantir que A e B sejam apresentados em posições laterais, representado pela fórmula A *sideof* B; ou que ambas tenham o mesmo tamanho, representado pela fórmula A *samesize* B. Dado que o leiaute espacial desse exemplo é estático, a validação espacial pode ser feita sobre a posição inicial das mídias do documento.

1.3.3.2. Caso 2

O exemplo do Caso 2 envolve a mudança ao longo do tempo do leiaute espacial de um documento multimídia. Neste exemplo, A tem uma posição fixa e B move-se ao longo da tela, mudando sua posição de maneira incremental ao longo de t_1 unidades de tempo. A Figura 1.7 apresenta o leiaute espaço-temporal percebido pelo espectador do documento d.

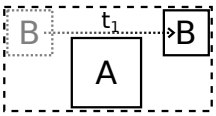


Figura 1.7: Leiaute espaço-temporal do caso 2

A validação do leiaute espaço-temporal do exemplo acima não é tão simples quanto validar o leiaute espacial e, em paralelo, validar o leiaute temporal. Por exemplo, suponha que o autor deseje garantir que, em algum ponto, enquanto se move pela tela, B irá se sobrepor a A. Tal verificação requer verificar se, em pelo menos um momento durante a execução do documento, A e B irão se sobrepor. Tal propriedade deve ser codificada pela composição de propriedades temporais e espaciais, tal como *somepoint*$(A$ *overlap* $B)$.

Nos exemplos apresentados anteriormente, (i) o leiaute espacial e temporal são independentes ou (ii) o leiaute espacial é dependente do temporal. Como visto acima, o exemplo apresentado no segundo caso é chamado de um leiaute espaço-temporal.

A validação puramente temporal atua somente no eixo temporal, verificando a consistência das relações temporais entre mídias. Por outro lado, a validação puramente espacial atua somente no eixo espacial, verificando a consistência das relações espaciais entre mídias. A validação espaço-temporal atua em ambos os eixos simultaneamente, considerando o caso onde mídias mudam sua posição ao longo do tempo.

1.4. Ciclo de Vida de Documentos Multimídia

A especificação contida em um documento multimídia pode variar desde sua criação até sua execução. Tal especificação pode variar de construções em um alto nível de abstração, seguindo uma linguagem de autoria, até eventos de execução em baixo nível de abstração. Diferentes trabalhos relacionados a documentos multimídia publicados na literatura comumente assumem um ciclo de vida composto pelas etapas de *concepção*, *armazenamento* e *execução* (trabalhos relacionados à autoria de documentos, como [Muchaluat-Saade 2003, Bulterman et al. 2013]) ou composto pelas etapas de *concepção*, *adaptação* e *execução* (trabalhos relacionados à adaptação de documentos, como [Lemlouma and Layaïda 2004, Na and Furuta 2001]).

Com base em tais abordagens, foi definido o ciclo de vida [dos Santos 2016] descrito na Figura 1.8. O ciclo apresentado é uma generalização de tais abordagens visando a identificação dos passos onde a validação de um documento é necessária. Ele foi projetado de modo que abordagens como [Sarkis et al. 2014, Lemlouma and Layaïda 2004, Laborie et al. 2011, Teixeira et al. 2012, Soares et al. 2012] possam ser representadas. Tais abordagens representam novas facilidades de apresentação de documentos multimídia, como a personalização da apresentação para um dado espectador [Lemlouma and Layaïda 2004, Laborie et al. 2011], a divisão da apresentação em múltiplos dispositivos [Sarkis et al. 2014] e a edição dinâmica da apresentação (por exemplo via anotação [Teixeira et al. 2012] ou edição ao vivo [Soares et al. 2012]).

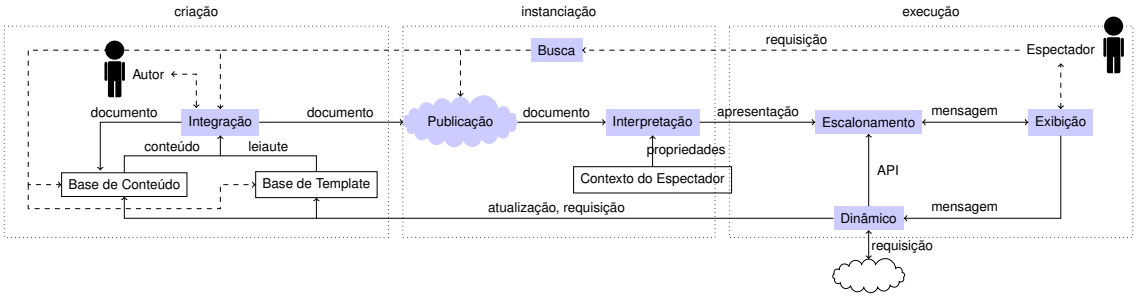


Figura 1.8: Ciclo de Vida de Documentos Multimídia

De acordo com a Figura 1.8, o ciclo de vida de um documento multimídia é composto de três fases principais: *criação*, *instanciação* e *execução*. Tais fases são representadas por retângulos pontilhados.

Cada fase é subdividida em passos (*Integração*, *Publicação*, *Interpretação*, *Escalonamento*, *Exibição*, *Dinâmico*) representados por retângulos pintados ou por uma nuvem pintada (como é o caso da *Publicação*). Armazenamento de dados é representado por retângulos vazios (como é o caso da *Base de Conteúdo*, *Base de Template* e *Contexto do Espectador*). Fluxos de dados são representados por linhas sólidas, enquanto linhas tracejadas representam uma informação sendo trocada com o *Espectador* ou com o *Autor*.

Referente à fase de *criação*, é importante ressaltar três aspectos. (i) Documentos são também considerados como conteúdo, isto é, podem também ser armazenados na base de conteúdo e usados na criação de novos documentos. (ii) Um template é considerado

como um leiaute genérico que é instanciado no passo de *Integração* para um determinado conteúdo. Assim, um template pode representar desde construções em uma determinada linguagem de autoria até especificações parciais de um documento que devem ser preenchidas com mídias específicas. (iii) Um documento pode ser criado tanto pelo autor quanto por um processo de geração automática. No primeiro caso, é possível que o autor use templates para facilitar o processo de criação de um documento, como discutido em [Damasceno et al. 2014]. No segundo caso, documentos são criados de acordo com uma requisição feita pelo espectador, combinando conteúdos existentes com templates predefinidos. É importante ressaltar que ainda assim o autor tem um importante papel na criação de tais templates.

De acordo com uma requisição feita pelo *Espectador*, no passo de *Integração* serão tomadas uma das seguintes ações: (i) serão organizados conteúdos disponíveis na *Base de Conteúdo* de acordo com o leiaute descrito em um template na *Base de Templates* ou (ii) se a requisição resultar em um documento já armazenado na *Base de Conteúdo*, o passo de *Integração* irá repassar tal documento adiante sem criar um novo. É importante notar que quando um novo documento é criado no passo de *Integração*, o documento resultante pode ser tanto guardado na *Base de Conteúdo* para uso futuro quanto usado como entrada para o passo de *Publicação*.

O passo de *Publicação* representa a transmissão[1] do documento para o espectador, de acordo com a requisição recebida. Um documento publicado é instanciado, no passo de *Interpretação*, em uma apresentação de acordo com o contexto do espectador. O contexto do espectador é composto de propriedades expressando (mas não limitadas a): (i) informação sobre o espectador, como idioma, localização, gênero, idade, etc; (ii) características do(s) dispositivo(s) de exibição do espectador, como tamanho de tela, tipos de mídias cuja exibição é suportada, disponibilidade de rede, etc; e (iii) preferências do usuário, como predisposição à interação, tipos de mídias preferidas, etc. Portanto, o documento executado deveria ser o mais adequado às características e preferências do espectador e dispositivos de exibição disponíveis.

O passo de *Escalonamento* mantém a sincronização da apresentação, exibindo cada conteúdo no tempo, espaço e dispositivo correto. O passo de *Exibição* atua como uma interface entre a apresentação e o espectador, repassando eventos de interação do espectador de volta para o passo de *Escalonamento*. Durante a execução, em relação a ocorrências de eventos (como por exemplo interação do espectador), o passo *Dinâmico* acessa APIs disponíveis no passo de *Escalonamento* para edição da apresentação. Exemplos de tais APIs são a API DOM (Document Object Model) [W3C 2000], a API SMIL DOM [W3C 1999], os Comandos de Edição ao Vivo de NCL [Soares et al. 2006] e a API Web Animations [W3C 2014]. Este passo pode ainda trocar informações com outros pares na rede, bem como armazenar novos conteúdos na *Base de Conteúdo* e novos templates na *Base de Templates*, ou requisitar tais informações destas bases.

[1]É por isso que o passo de *Publicação* é representado por uma nuvem, diferentemente dos demais passos.

1.4.1. Casos de Uso

Usualmente, ao se referir ao ciclo de vida de documentos multimídia, a literatura o divide em dois lados: o lado *servidor/difusor* e o lado *cliente/espectador*. No lado *cliente* (ou *espectador*), ficam os passos relacionados à execução do documento. O cliente pode ser tão complexo quanto possível - por outro lado tão simples quanto possível - relativo aos passos por ele realizados. O lado *servidor* (ou *difusor*) fica com os passos restantes. Quanto mais simples é o cliente, mais complexo é o servidor e vice-versa.

Uma divisão comum em ambientes de TV digital e IPTV [ETSI 2008, ATSC 2009, ARIB 2014, ITU 2009] e ainda na web, é usar o passo de *Publicação* como a divisão entre os lados servidor e cliente. O trabalho apresentado em [Lemlouma and Layaïda 2004] apresenta uma divisão diferente, seguindo a ideia de um cliente mais simples, onde somente o passo de *Exibição* está no cliente. Em tal cenário, um documento é criado e sua execução é tratada pelo servidor. O cliente recebe amostras das mídias a serem apresentadas, para um dado momento, e comunica eventos da apresentação e interação do espectador de volta para o servidor.

Esta seção discute trabalhos que implementam partes do ciclo de vida apresentado na Figura 1.8. Os trabalhos apresentados cobrem diferentes áreas de pesquisa em multimídia e inspiraram o ciclo de vida aqui apresentado. Os parágrafos a seguir apresentam tais trabalhos junto com uma discussão sobre os passos que eles implementam.

O trabalho em [Sarkis et al. 2014] apresenta uma abordagem para a divisão de uma aplicação multimídia em duas telas. Este trabalho se encaixa no ciclo de vida apresentado na Figura 1.8 como segue. No passo de *Integração*, elementos do documento são anotados com informação sobre a tela onde serão apresentados, de acordo com a requisição recebida. O documento anotado é enviado, no passo de *Publicação*, para ambos os clientes identificados na requisição recebida. A seguir, na etapa de *Interpretação*, um dado cliente divide o documento mantendo somente os elementos a serem por ele apresentados visíveis e "escondendo" os demais. Durante a execução, o passo *Dinâmico* recebe a interação do espectador com a apresentação e mantém comunicação entre ambas as telas de forma que o documento como um todo mantenha sua sincronização.

O trabalho apresentado em [Laborie et al. 2011] propõe uma abordagem para a adaptação no cliente de documentos multimídia. Tal abordagem implementa o passo de *Interpretação* de forma que um dado documento seja traduzido em um modelo abstrato, como um conjunto de objetos e relações espaço-temporais entre eles. Relações neste modelo são representadas por restrições. Além das relações, restrições são também utilizadas para representar diretrizes do autor. O processo de adaptação combina restrições do documento com outras representando o *Contexto do Espectador*. Dado que o conjunto de restrições resultante desta combinação seja satisfatível, o documento pode ser adaptado sem precisar de modificações. Caso contrário, relações no documento devem ser modificadas até que o conjunto de restrições seja satisfatível.

Ambos os trabalhos apresentados em [Teixeira et al. 2012] e [Soares et al. 2012] atuam no passo *Dinâmico*. O trabalho em [Teixeira et al. 2012] propõe uma abordagem para a anotação de documentos e o trabalho em [Soares et al. 2012] propõe uma abordagem para a adaptação no lado cliente de um documento.

Em [Teixeira et al. 2012], o passo *Dinâmico* é implementado como uma extensão do middleware cliente de forma a permitir ao espectador inserir texto, som ou imagens em uma apresentação. A abordagem proposta captura eventos de interação do espectador para criar anotações sobre o conteúdo sendo apresentado. As anotações produzidas são sincronizadas com o conteúdo principal sendo apresentado através de um documento NCL. Apesar de não ser explorado no artigo, o documento resultante, isto é, contendo as anotações criadas pelo espectador, pode ser enviado de volta para a *Base de Conteúdo* para que seja provido para outros espectadores em futuras visualizações do mesmo conteúdo ou mesmo compartilhado com outros espectadores assistindo ao mesmo conteúdo.

Em [Soares et al. 2012], o passo *Dinâmico* é implementado usando um script Lua que, reagindo a eventos de interação do espectador, dispara requisições para a *Base de Conteúdo* de forma a obter novos vídeos relacionados ao conteúdo sendo apresentado em um dado momento. Tal abordagem cria um novo documento com o novo conteúdo a ser incluído na apresentação, comparando a nova e a atual versão da apresentação para determinar as mudanças a serem realizadas visando atualizar a apresentação. Em seguida são enviados comandos de edição ao vivo de NCL [Soares et al. 2006] para realizar as modificação ao longo da execução.

1.4.2. Validação ao Longo do Ciclo

Visando manter a consistência de um documento ao longo do seu ciclo de vida, a validação é importante em diferentes etapas.

Inicialmente, é importante garantir que conteúdos e template(s) combinados no passo de *Integração* resultem em um documento consistente. Isto pode ser obtido realizando a validação do documento no passo de *Integração*. Tal validação leva em consideração propriedades associadas aos conteúdos e templates utilizados, assim como propriedades relacionadas ao documento sendo criado. Estas propriedades relacionadas ao documento definem o que nos referimos como diretrizes do autor. Validação no passo de *Integração* deve ser capaz de suportar diferentes paradigmas de autoria de documentos multimídia (relações, eventos, composições, etc). É importante mencionar, entretanto, que algumas linguagens multimídia declarativas provêem a possibilidade de se utilizar linguagens de script auxiliares (como Lua junto com NCL e JavaScript junto com HTML) para implementar facilidades não suportadas pela linguagem em uso. A validação de tais scripts, entretanto, foge ao escopo da validação da linguagem declarativa em uso.

De acordo com o paradigma utilizado no passo de *Integração* é possível validar o documento ao mesmo tempo em que ele é descrito. Por exemplo, em uma abordagem baseada em restrições, como em [Jourdan et al. 1998], relações podem ser validadas enquanto são criadas. Por outro lado, em uma abordagem baseada em eventos, tal como [dos Santos et al. 2015], a validação pode ter de esperar a criação de um conjunto de relações antes de ser realizada. Nesse ponto do ciclo de vida do documento, a validação tem um papel *preventivo*, investigando se um documento pode ser instanciado e, uma vez apresentado, não leva a erros de execução. No caso de um erro ser encontrado, a ferramenta de validação pode tanto (i) "corrigir" automaticamente o documento ou (ii) dar um *feedback* ao autor para que este possa "corrigir" o documento sendo criado. Ainda, a validação neste ponto também é *parcial*, visto que o conteúdo (ou mesmo o leiaute) de

um documento pode mudar dinamicamente durante execução.

Após a fase de *criação*, a validação de um documento ao longo do seu ciclo de vida deve sempre retornar uma apresentação executável, ainda que através de alguma forma automática de "correção de erro". Erros após a fase de criação podem interromper a apresentação do documento requisitado. Apesar de *feedbacks* serem importantes para que o autor possa corrigir novas versões do documento, tal prática não ajuda na correção do documento sendo apresentado. Ainda, o tempo de resposta da validação tem um papel importante, para evitar que o espectador espere um longo tempo para começar a apresentação do documento (*Interpretação*) ou trave durante sua execução (*Escalonamento*).

Durante o processo de instanciação no passo de *Interpretação*, é importante garantir que um documento criado possa ser instanciado em uma apresentação levando em conta o contexto do espectador. Isso pode ser realizado validando a combinação tanto de propriedades associadas ao documento quanto outras expressando o contexto do espectador. De acordo com a forma como o documento é descrito e dado que mais de uma apresentação pode ser instanciada a partir de um documento, a validação do documento pode ser vista como uma forma de prover personalização e ajudar a adaptação do documento, escolhendo a instância de apresentação mais adequada ao contexto do espectador. Ainda, dado que tanto a adaptação quando a validação podem ser realizadas usando as mesmas técnicas, nos referimos a adaptação de um documento como um caso especial de validação.

Uma vez que um documento esteja sendo executado, ele pode evoluir seja por interação do espectador quanto por mudanças incrementais feitas no passo *Dinâmico*. Similar à escrita colaborativa [Sun et al. 1998], nesse ponto, é importante prover algum tipo de *preservação de intenção*, para evitar que a apresentação se torne incoerente em relação às diretrizes do autor. Da mesma forma como feito no passo de *Interpretação*, na etapa de edição *Dinâmica*, a validação deve ser capaz de combinar propriedades representando características da apresentação com outras representando as mudanças incrementais. Tal validação deve sempre resultar em uma apresentação executável, caso contrário, uma mudança incremental deve ser ignorada para evitar que a apresentação se torne inconsistente.

É importante salientar que diretrizes do autor são propriedades a serem preservadas ao longo do ciclo de vida de um documento. Devido a personalização (no passo de *Interpretação*) ou mudanças incrementais (no passo *Dinâmico*), relações expressando o leiaute do documento podem ser modificadas. Diretrizes do autor representam uma forma de guiar tais modificações de forma a prover *preservação de intenção* [Sun et al. 1998] ao longo do ciclo de vida.

1.5. Soluções Propostas na Literatura

A literatura é rica em trabalhos sobre a validação de documentos multimídia. As soluções aqui apresentadas são classificadas de acordo com o método aplicado na validação de um documento. O primeiro grupo de trabalhos [Felix 2004, Gaggi and Bossi 2011, Bouyakoub and Belkhir 2011, Júnior et al. 2012, King et al. 2004, dos Santos 2016] realiza a validação de um documento investigando seu estado ao longo da sua execução. Tal abordagem pode ser feita analisando a alcançabilidade de estados, ou pela aplicação de axiomas sobre o estado do documento. O segundo grupo de trabalhos

[Bertino et al. 2005, Laborie et al. 2011, Elias et al. 2006, dos Santos 2016] realiza a validação de um documento checando a consistência de um conjunto de restrições.

1.5.1. Trabalhos Referentes a Alcançabilidade de Estados

Felix em [Felix 2004] apresenta uma abordagem para a validação temporal de documentos NCL [ITU 2009] através da aplicação de técnicas de *model-checking*. Ele apresenta uma notação para a descrição das características temporais de NCL, que segue um modelo baseado em eventos, a qual é transformada em uma representação similar a uma máquina de estados indicando o leiaute temporal de um documento. A transformação cria uma máquina de estados para cada mídia em um documento e uma "máquina de estados de sincronização" representando cada relação declarada no documento. Tais máquinas de sincronização são usadas para relacionar a ocorrência de eventos nas máquinas representando as mídias de um documento. A validação de um documento NCL é então realizada sobre um autômato resultante da combinação de tais máquinas de estado, o qual representa o leiaute temporal do documento. A validação é feita através de fórmulas em lógica temporal definidas pelo autor.

Gaggi e Bossi em [Gaggi and Bossi 2011] definem uma semântica formal para a linguagem SMIL através de um conjunto de regras de inferência inspiradas na lógica de Hoare. SMIL segue um modelo baseado em sincronização hierárquica e as regras descrevem o estado do documento antes e depois da execução de uma construção da linguagem SMIL. Assim, na fase de autoria, a estrutura de um documento SMIL é enriquecida com assertivas expressando propriedades temporais. Uma outra aplicação da semântica definida é o conceito de equivalência entre construções. Tal conceito garante que dois conjuntos de construções da linguagem SMIL podem ser substituídos em um documento sem mudar a apresentação resultante. A validação de um documento é realizada pela aplicação de axiomas, também definidos na semântica proposta, que verificam se uma dada construção, ou um conjunto delas, corretamente modifica o estado do documento. Caso contrário, a solução apresenta o problema encontrado para que o autor possa corrigi-lo.

Bouyakoub e Belkhir em [Bouyakoub and Belkhir 2011] apresentam uma ferramenta de autoria incremental para documentos SMIL, chamada *SMIL Builder*. Sempre que o autor realiza alguma mudança no documento, SMIL Builder verifica se tal modificação tornará o leiaute temporal do documento inconsistente. Em caso positivo, a mudança é rejeitada e um relato do erro é apresentado ao autor. Tanto a autoria incremental quanto a verificação de consistência são suportadas pelo modelo H-SMIL-Net [Bouyakoub and Belkhir 2008]. H-SMIL-Net estende Redes de Petri de forma a poder representar de forma completa conceitos temporais da linguagem SMIL. Inconsistências são detectadas seja por construção, por exemplo, quando uma transição tem mais arcos do que possível, ou por verificação, no modelo, das transições disparadas.

Júnior et al. em [Júnior et al. 2012] usam uma abordagem dirigida a modelos para a validação de documentos NCL. A validação é realizada transformando um documento NCL em uma estrutura de Kripke. Esta transformação é feita em duas etapas. Na primeira etapa o documento é representado em uma linguagem chamada FIACRE, como um conjunto de componentes e processos (representando o comportamento de um componente). A segunda etapa transforma a representação em FIACRE para uma estrutura de Kripke.

A validação usa uma ferramenta de *model-checking* e fórmulas em lógica temporal representando as propriedades a serem validadas. A validação espacial é brevemente discutida em [Júnior et al. 2012] e é realizada sobre a posição inicial das mídias declaradas no documento. Como apresentado na Tabela 1.1, esta abordagem cobre tanto as dimensões temporal e espacial, entretanto a dimensão espacial é estática.

King et al. em [King et al. 2004] definem extensões para a linguagem SMIL permitindo que autores possam descrever como o leiaute espaço-temporal deve reagir a um evento. Mudanças na posição e tamanho são descritas por um conjunto de expressões, as quais podem levar em consideração o estado do documento. O trabalho apresenta uma abordagem para calcular durante a execução do documento o valor de tais expressões e, portanto, fazer a mudança no leiaute espacial de acordo com a expressão. Tal abordagem, apesar de não ter por objetivo validar o leiaute temporal, ou espaço-temporal, de um documento, pode ser considerada um caso especial de validação, onde o leiaute da aplicação é recalculado dada a passagem do tempo ou a ocorrência de um evento.

dos Santos et al. em [dos Santos 2016] apresentam uma abordagem para a validação de documentos multimídia, especificados tanto em NCL quanto em SMIL, tendo como base um modelo formal chamado *SHM*. Uma das técnicas de validação propostas em [dos Santos 2016], chamada *RWT*, captura o comportamento de um documento representado no modelo *SHM* em termos do seu estado ao longo da sua execução. A validação proposta é baseada em uma teoria de reescrita $\mathscr{R}_{RWT}$, capaz de representar o comportamento de documentos multimídia ao longo do tempo e espaço. A teoria $\mathscr{R}_{RWT}$, induz um sistema de transição $\mathscr{S}_{RWT}$ onde os estados representam o estado do documento em um dado instante e transições representam a passagem do tempo ou uma interação do espectador. A validação tanto de propriedades temporais quanto espaciais é realizada através *model-checking* sobre $\mathscr{S}_{RWT}$. É importante destacar que em [dos Santos 2016] a validação de um documento pode ser realizada com base em um conjunto de propriedades comuns, bem como em propriedades construídas pelo autor. Tal facilidade é provida através de um conjunto de propriedades atômicas que podem ser combinadas formando as diretrizes desejadas pelo autor.

1.5.2. Trabalhos Referentes a Validação de Restrições

Bertino et al. em [Bertino et al. 2005] propõem um modelo de autoria baseado em restrições. Um documento neste modelo consiste de um conjunto de tópicos, onde cada tópico é composto de mídias semanticamente relacionadas. O sistema automaticamente agrupa mídias em tópicos de acordo com as restrições definidas pelo autor. O processo de geração da apresentação é responsável por três tarefas principais, são elas: checagem de consistência, geração da estrutura da apresentação e geração dos tópicos. O sistema adiciona no conjunto de restrições, outras restrições que embora não tenham sido definidas explicitamente, são consequência das restrições definidas pelo autor. A checagem de consistência é então realizada sobre esse conjunto de restrições. Se uma inconsistência é obtida, o sistema aplica técnicas de relaxamento para reduzir o conjunto de restrições em um conjunto consistente. Quando a redução automática não é possível, a revisão do autor é requisitada. O processo de criação da estrutura da apresentação cria um grafo dirigido que representa a estrutura da apresentação. Cada vértice do grafo representa um tópico e arestas representam uma conexão entre os tópicos. Após esta etapa, o sistema associa

mídias aos tópicos, cria o leiaute espacial e a sequência temporal de mídias para cada tópico.

Laborie et al. em [Laborie et al. 2011] apresentam uma abordagem de adaptação do leiaute de um documento de acordo com o dispositivo de exibição. No artigo, a linguagem SMIL é utilizada para demonstrar a abordagem proposta. A abordagem proposta cria uma descrição abstrata do documento como um conjunto de objetos e restrições representando relações temporais e espaciais entre objetos. É ainda considerado um perfil contendo restrições do dispositivo de exibição e preferências do usuário. De acordo com o conjunto de potenciais execuções de um documento $\mathscr{M}_s$ dado pela representação abstrata do documento, e o conjunto de potenciais execuções $\mathscr{M}_p$ dado pelo perfil, o processo de adaptação calcula a interseção $\mathscr{M}_s \cap \mathscr{M}_p$ para determinar se alguma adaptação é necessária. No caso de o documento precisar ser adaptado, o objetivo da solução proposta é alterar relações declaradas no documento de forma que este fique aderente ao perfil e que a distância (do comportamento) em relação ao documento original seja mínima.

Elias et al. em [Elias et al. 2006] também propõem um modelo de autoria baseado em restrições. A abordagem proposta define dois operadores *TEMPORAL* e *SPATIAL*, para modelar relações temporais e espaciais, respectivamente. Cada operador permite que o autor defina um valor de prioridade. Para manter a consistência do conjunto de restrições, sempre que necessário, restrições são removidas de acordo com seu valor de prioridade. No caso de duas restrições inconsistentes apresentarem o mesmo valor de prioridade, técnicas de relaxamento são aplicadas para determinar a restrição a ser removida. A checagem de consistência é feita achando a árvore geradora mínima T do conjunto de restrições. Restrições que criam ciclos são removidas para manter a natureza acíclica de T. A checagem de completude é feita buscando todas as mídias que são alcançadas a partir da primeira mídia. Se essa busca retorna o conjunto de vértices de T, então todas as mídias são alcançadas direta ou indiretamente a partir da mídia inicial. Caso contrário, o autor tem que definir restrições para tornar o conjunto de restrições completo. Com o uso do operador *SPATIAL*, é possível determinar se duas mídias A e B se sobrepõem. Cada restrição espacial é associada a um intervalo, de forma que uma restrição espacial deve ser satisfeita dentro deste intervalo. Apesar de o trabalho apresentado em [Elias et al. 2006] não definir atributos espaciais (posição e tamanho) em função do tempo, ele representa um exemplo de restrições parametrizadas pelo tempo.

dos Santos et al. em [dos Santos 2016] em sua abordagem baseada no modelo formal *SHM* provêem uma segunda técnica de validação, chamada *SMT*, que captura o comportamento de um documento em termos de intervalos e ocorrências de eventos. Intervalos temporais e regiões espaciais representando fragmentos do documento e relações espaço-temporais entre tais intervalos/regiões são descritas através de fórmulas em SMT (*Satisfiability Modulo Theories*) [De Moura and Bjørner 2011]. A validação tanto de propriedades temporais quanto espaciais é realizada através de um *solver* SMT. Assim como na técnica anteriormente descrita, a validação pode ser realizada com base em um conjunto de propriedades comuns, bem como em propriedades construídas pelo autor.

1.5.3. Comparação entre as Soluções Propostas

A seção anterior apresentou algumas soluções propostas na literatura para a validação de documentos multimídia. Tais soluções foram classificadas de acordo com o método aplicado na validação e o tipo de validação realizada (temporal e/ou espacial). A Tabela 1.1 resume tais soluções de acordo com as três principais fases do ciclo de vida onde atuam (*criação*, *instanciação* e *execução*).

	Temporal	Temporal + Espacial	Espaço-temporal	Alcance de Estados	Validação de Restrições	Criação	Instanciação	Execução
[Felix 2004]	✓			✓		✓		
[Gaggi and Bossi 2011]	✓			✓		✓		
[Bouyakoub and Belkhir 2011]	✓			✓		✓		
[Júnior et al. 2012]		✓		✓		✓		
[King et al. 2004]			✓	✓				✓
[Bertino et al. 2005]		✓			✓	✓	✓	
[Laborie et al. 2011]		✓			✓	✓	✓	
[Elias et al. 2006]			✓		✓	✓		
[dos Santos 2016]			✓	✓	✓	✓		✓

Tabela 1.1: Comparação entre as soluções propostas na literatura

Como pode ser visto na Tabela 1.1, as soluções apresentadas em [Felix 2004, Gaggi and Bossi 2011, Bouyakoub and Belkhir 2011] apresentam uma abordagem puramente temporal, onde a validação de um documento é realizada investigando seu estado ao longo de sua execução. Em [Felix 2004], a validação é feita analisando a alcançabilidade de estados, em [Bouyakoub and Belkhir 2011] a validação é feita verificando a consistência da rede de Petri representando um documento e em [Gaggi and Bossi 2011] analisando se o estado do documento muda de acordo com alguns axiomas. A validação do leiaute espacial do documento não é discutida nesses trabalhos.

As soluções apresentadas em [Júnior et al. 2012, Bertino et al. 2005, Laborie et al. 2011] cobrem tanto as dimensões temporais e espaciais. No primeiro, a validação de um documento é realizada via *model-checking* e, no segundo e terceiro trabalhos a validação é realizada pela checagem de consistência de um conjunto de restrições. Entretanto, em todos esses trabalhos, a dimensão espacial é estática, visto que restrições espaciais não mudam ao longo do tempo. Ainda, raciocinar sobre tempo e espaço é feito como dois problemas separados.

Finalmente, as soluções propostas em [Elias et al. 2006, dos Santos 2016] provêem uma validação verdadeiramente espaço-temporal, dado que relações espaciais podem variar ao longo do tempo. Apesar de [King et al. 2004] suportar mudanças no leiaute

espacial ao longo do tempo, tal abordagem não tem por objetivo suportar a validação do documento. Em [Elias et al. 2006] a validação é realizada pela checagem de consistência de um conjunto de restrições. Em [dos Santos 2016], ambas as técnicas são suportadas.

As soluções apresentadas nesta seção atuam em diferentes etapas do ciclo de vida de um documento (Figura 1.8), como visto na Tabela 1.1. Grande parte das soluções atuam na etapa de *criação*, como esperado, uma vez que propõem ferramentas ou abordagens para facilitar a autoria de documentos. Ambas as soluções [Bertino et al. 2005, Laborie et al. 2011] levam em consideração o contexto do espectador para prover adaptação do documento antes de sua execução, portanto atuando na etapa de *instanciação*. A validação provida por tais trabalhos, entretanto, possui um papel preventivo. Eles investigam se um documento é executável e se nenhuma inconsistência pode surgir durante sua execução. Tais soluções, entretanto, não levam em consideração o caso em que um documento é editado dinamicamente. É importante salientar também que apesar de tais propostas focarem na adaptação, elas poderiam ser utilizadas também na etapa de criação para validar um documento sendo criado.

As soluções apresentadas em [King et al. 2004, dos Santos 2016] são os únicos trabalhos que atuam na etapa de *execução*. Em [King et al. 2004], é proposta uma forma de calcular, durante a execução, o leiaute espacial de um documento. Em [dos Santos 2016], são providas duas técnicas de validação, onde ambas atuam na etapa de *criação*. Ainda, um protótipo da validação usando restrições para a validação de um documento durante sua execução também é apresentado.

1.6. Exemplo de Verificação de Consistência

Esta seção apresenta um documento multimídia para exemplificar a validação discutida ao longo deste capítulo. O documento de exemplo, chamado “Roteiro do Dia” é um documento especificado com a linguagem NCL [ITU 2009] e está disponível para *download* no Clube NCL[2].

Este documento é um exemplo de programa de televisão interativo. O programa em questão faz uma visita turística a uma cidade, apresentando seus principais pontos turísticos. No programa, a cidade do Rio de Janeiro é apresentada, sendo visitados os seguintes pontos turísticos: Central do Brasil, Copacabana, Jardim Botânico e a Gafieira Estudantina.

O espectador pode escolher, ao início da apresentação do documento se está disposto ou não a interagir. Estando disposto, ao fim da visita a um ponto turístico, o espectador escolhe o próximo ponto a ser visitado. Assim, uma visita turística personalizada é apresentada. Caso o espectador não esteja disposto a interagir, um roteiro de visita escolhido previamente pelos autores do programa é apresentado.

Nesta seção, será utilizada como exemplo de solução para a validação do documento acima, a API aNaa (API NCL de Autoria e Análise) [dos Santos 2012, dos Santos 2016]. A Figura 1.9 apresenta aNaa4Web, uma interface web construída sobre aNaa para utilizar a validação por ela provida.

[2]http://clube.ncl.org.br

Figura 1.9: Interface inicial da ferramenta aNaa4Web

Uma vez escolhido o documento a ser validado, a ferramenta apresenta em sua barra superior, o identificador e o caminho para o arquivo principal do documento. A ferramenta ainda identifica as mídias que possuem uma duração implícita (áudio ou vídeo) para que o usuário indique a duração em segundos de cada. A Figura 1.10 apresenta a interface da ferramenta após a escolha do documento "Roteiro do Dia".

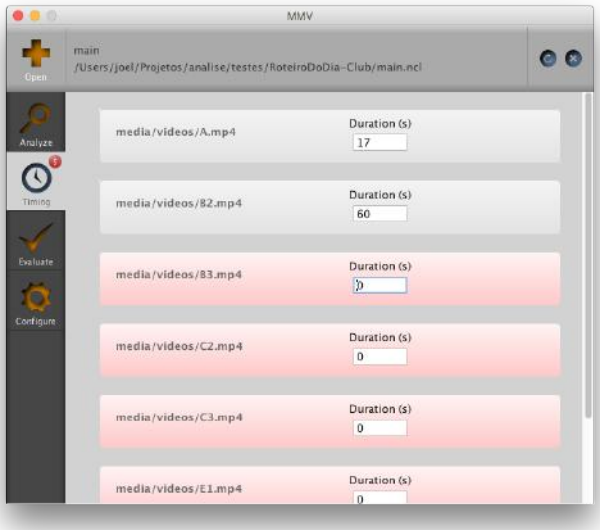


Figura 1.10: Interface para indicação da duração das mídias

Na Figura 1.10 é possível identificar o contador apresentado sobre o botão "Timing". Este contador indica o número de mídias para as quais a duração ainda não foi identificada. Ainda, para cada mídia sem duração, uma campo é apresentado onde o usuá-

rio poderá indicar a duração da mídia. Uma vez indicada a duração, o campo muda da cor vermelha para a cinza e o contador é decrementado.

Ao clicar no botão "Analyze", um menu é apresentado onde o usuário pode escolher se quer realizar a validação estrutural ou comportamental de um documento. Inicialmente será apresentado o uso da ferramenta para a validação estrutural do documento. A validação estrutural é realizada com base nas propriedades estruturais apresentadas na Seção 1.3.1. Cada propriedade é instanciada para a linguagem NCL, gerando um conjunto de propriedades a serem validadas. O resultado da validação é apresentado na Figura 1.11.

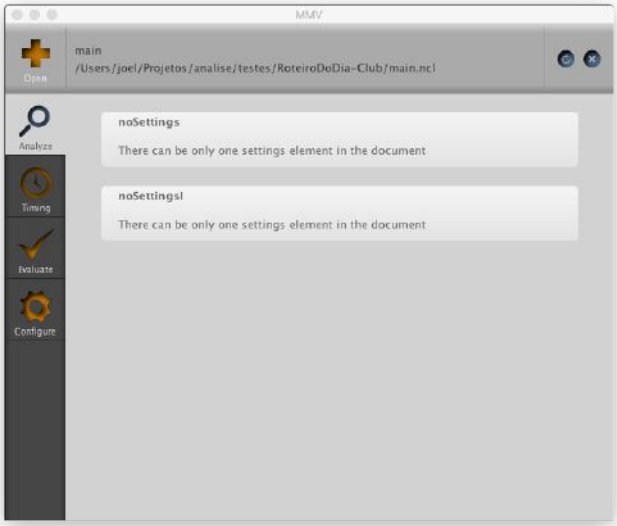


Figura 1.11: Resultado da Validação Estrutural

Note que, duas mensagens de erro são apresentadas, ambas relacionadas ao fato de existirem duas mídias do tipo *settings* no documento. Uma mídia do tipo *settings* é uma mídia que representa um conjunto de variáveis globais em um documento e, de acordo com a especificação da linguagem NCL [ABNT 2011], deve ser única.

Uma vez corrigido o documento, pode ser realizada sua validação comportamental. A API aNaa provê uma validação espaço-temporal do comportamento do documento, seguindo uma abordagem baseada na alcançabilidade de estados. Como apresentado na Seção 1.5.1, o documento a ser validado é representado no modelo formal *SHM* como uma teoria de reescrita. Esta teoria induz um sistema de transição representando as possíveis execuções do documento Roteiro do Dia, onde cada estado representa o estado das mídias declaradas no documento e as transições representam uma progressão temporal ou interação do espectador. A interação, nesse caso, representa a escolha da próxima atração turística a ser visitada.

A validação comportamental é realizada através *model-checking* com base nas propriedades comportamentais gerais apresentadas na Seção 1.3.2. Cada propriedade é instanciada, para cada mídia do documento, em um comando em lógica temporal e validada sobre o sistema de transição representando as possíveis execuções do documento.

A Figura 1.12 apresenta o resultado da validação comportamental do documento Roteiro do Dia.

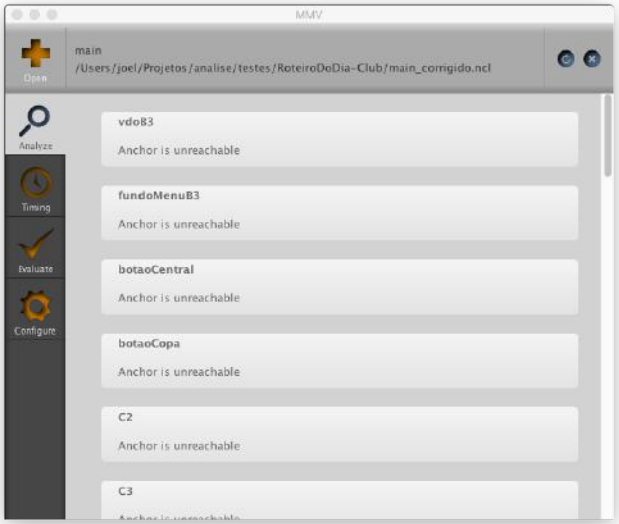


Figura 1.12: Resultado da Validação Comportamental

É possível notar que a ferramenta identifica diversos *warnings* no documento. Tais *warnings* estão relacionados ao fato de que diferentes mídias podem não ser apresentadas, de acordo com a interação do espectador.

Apesar da API aNaa suportar a validação de propriedades criadas pelo autor, a interface aNaa4Web só prevê a validação das propriedades gerais apresentadas na Seção 1.3.2. Uma segunda ferramenta construída tendo como base a API aNaa, chamada NCL-Tester [Barreto et al. 2016] permite que o autor crie um conjunto de propriedades específicas a um documento e realize sua validação. A interface do NCL-Tester é apresentada na Figura 1.13.

NCL-Tester define um conjunto de propriedades temporais que podem ser utilizadas como base para a criação de testes pelo autor. Para cada propriedade instanciada, o autor indica as mídias a serem consideradas. A validação das propriedades criadas pode ser feita em separado ou em conjunto.

No exemplo apresentado na Figura 1.13 a ferramenta NCL-Tester foi utilizada para criar um teste sobre o documento "Roteiro do Dia". O teste criado especifica que a apresentação do vídeo *vdoB3* (apresenta a cidade e chama o espectador a interagir) deve preceder a apresentação do vídeo *C2* (apresenta a Central do Brasil).

O teste criado é do tipo *meets*, sendo baseado na relação de Allen [Allen 1983] de mesmo nome. Na aba criada para o teste, a mídia *vdoB3* foi arrastada para a posição da mídia mestre do teste enquanto a mídia *C2* foi arrastada para a posição escrava. O teste é então traduzido para uma propriedade em lógica temporal e enviado para a API aNaa para que ela possa realizar a validação da propriedade. O resultado do teste é apresentado no painel do lado direito da tela. Além disso, a ferramenta indica visualmente o resultado do

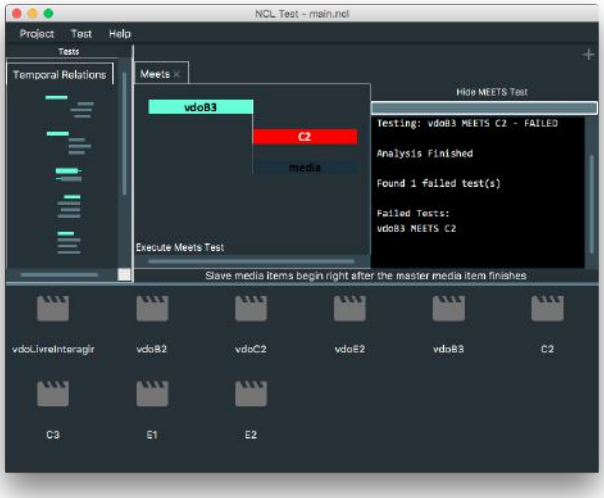


Figura 1.13: Interface do NCL-Tester [Barreto et al. 2016]

teste, alterando a cor da mídia escrava para verde, caso o teste passe, ou vermelho, caso falhe. O teste apresentado na Figura 1.13 falha, pois é possível que o vídeo *C2* não seja executado após o vídeo *vdoB3* devido a interação feita pelo espectador.

Esta seção apresentou um exemplo de documento NCL e sua validação usando uma solução presente na literatura. A Seção a seguir discute desafios de pesquisa ainda em aberto relacionados à validação de documentos multimídia.

1.7. Desafios de Pesquisa

Foram apresentadas na Seção 1.5 diferentes soluções para a validação de documentos multimídia disponíveis na literatura. Apesar da existência de diferentes soluções, algumas facilidades relacionadas à validação ainda são desafios em aberto.

Um primeiro desafio está relacionado à utilização da validação nas diferentes etapas do ciclo de vida de um documento. Grande parte dos trabalhos focam na validação durante a fase de *criação*. A validação durante as fases de *instanciação* e *execução* ainda não são muito exploradas na literatura. Além disso, uma abordagem para a definição de diretrizes pelo autor, que estejam disponíveis em tempo de execução, ainda não foi apresentada.

Documentos multimídia podem ser criados através do uso de templates de documentos [dos Santos and Muchaluat-Saade 2012, Neto et al. 2012, Deltour and Roisin 2006]. Um template representa uma especificação parcial de um documento que deve ser completada com mídias específicas escolhidas pelo autor. Visto que um template pode ser utilizado para a criação de uma gama de documentos compartilhando características comuns especificadas no template, realizar a validação de um template representa uma forma de garantir que documentos criados a partir deste template sejam consistentes.

Templates abstraem o conteúdo de um documento representando um conjunto de mídias semanticamente relacionadas como um componente do template. Relações podem

então ser definidas (i) entre componentes (conjuntos) ou (ii) sobre um mesmo componente (conjunto). Como um exemplo do primeiro caso, suponha um template definindo uma relação onde sempre que uma mídia do conjunto A é selecionada (componente foto), uma outra mídia do conjunto B (componente descrição) é apresentada. Um caso real seria uma aplicação com fotos e suas descrições, por exemplo. Como um exemplo do segundo caso, suponha um template definindo uma relação sobre um conjunto de mídias (componente foto), onde assim que uma mídia na posição i termina sua apresentação, começa a apresentação de uma mídia na posição $i+1$. Um caso real seria uma aplicação que apresenta um conjunto de fotos em sequência.

Um desafio para a validação de templates, portanto, é a representação destes dois casos. No caso em que uma relação é definida entre dois conjuntos, cada conjunto pode ser pensado como uma única mídia. No caso em que uma relação é definida sobre um mesmo conjunto, entretanto, a abordagem anterior não pode ser utilizada, sendo necessário instanciar esse template para um determinado número de mídias. Tal diferença de representação se apresenta como um grande desafio para a validação de templates.

Documentos multimídia podem ter parte de seu conteúdo obtido sob demanda quando executado. Nesse caso, atrasos na obtenção de tais conteúdos podem impactar na apresentação de um documento. Ainda, é possível que a plataforma onde um documento é executado não seja capaz de manter uma exata sincronização entre mídias, por exemplo, adicionando um pequeno atraso no início de apresentação de cada mídia. Tais características dependem da plataforma onde um documento é executado. É possível que um documento consistente torne-se inconsistente durante sua execução, devido a tais características. Um desafio de pesquisa é pensar em como representar tais características, simulando um mau funcionamento da plataforma de exibição, visando checar a robustez de um documento. Isto significa que o documento possui relações capazes de corrigir pequenos erros na sincronização do documento, mesmo quando alguma inconsistência é inserida pela plataforma de exibição.

Similar ao caso acima, é a validação de documentos multimídia distribuídos. No caso em que diferentes dispositivos são utilizados para apresentar um mesmo documento, devem ser levados em consideração, durante a validação, eventuais atrasos de comunicação entre os dispositivos responsáveis pela exibição de um documento. Exemplos de exibição de um documento em múltiplos dispositivos são aplicação com segunda tela, e aplicações mulsemídia (*multiple sensorial media*) [Ghinea et al. 2014] onde o espectador recebe estímulos sensoriais para outros sentidos além de visão e audição, já presentes em documentos multimídia. Tais estímulos são produzidos pelos chamados atuadores, que devem ser controlados em conjunto como em uma aplicação em múltiplos dispositivos.

Por fim, um outro desafio de pesquisa é no caso de aplicações mulsemídia. O desafio, nesse caso é como representar estímulos sensoriais e sensores, de forma a poder-se validar um documento que leve em consideração tais componentes.

1.8. Conclusão

Um documento multimídia descreve um conjunto de mídias e como estas devem ser apresentadas, tanto no tempo, quanto no espaço. Quando executada, essa descrição induz um arranjo particular das mídias no tempo e espaço, chamado de uma apresentação multimí-

dia. A descrição contida em um documento é comumente textual, utilizando uma linguagem de autoria. Diferentes linguagens de autoria, bem como trabalhos publicados na literatura provêem formas de se adaptar um documento ao contexto do leitor. Por exemplo, é possível reorganizar mídias na tela do dispositivo de exibição de acordo com seu tamanho. Algumas linguagens provêem ainda a possibilidade de editar dinamicamente um documento ao longo da sua apresentação. Isso significa que relações espaço-temporais em um documento podem ser modificadas enquanto o documento é executado.

Desde sua criação até sua execução, um documento passa por um conjunto de etapas, chamadas aqui de o ciclo de vida de um documento multimídia. Visto que a especificação contida em um documento pode variar ao longo do seu ciclo de vida, é importante verificar, ao longo de cada etapa, se modificações realizadas em um documento o transformam em algo diferente do que foi especificado em tempo de criação. Assim, dizemos que um documento multimídia é consistente, se este se mantém aderente a um conjunto de diretrizes do autor. A verificação da consistência de um documento é feita através de sua validação.

Diferentes tipos de validação podem ser feitos sobre um documento. Neste capítulo foram discutidas a validação estrutural e comportamental de um documento. A validação comportamental foi ainda classificada em três tipos: puramente temporal, temporal e espacial analisadas em separado e espaço-temporal. Este capítulo apresentou ainda o ciclo de vida de um documento multimídia composto de etapas onde a validação de um documento se faz necessária. Foram ainda apresentadas soluções propostas na literatura e como cada solução se encaixa no ciclo apresentado e tipo de validação provida. Uma das soluções apresentadas foi utilizada para apresentar um exemplo de verificação de consistência, demonstrando seu uso para validar um documento em diferentes etapas do seu ciclo de vida. Por fim, foram discutidos alguns desafios de pesquisa relacionados à validação de documentos multimídia ainda em aberto.

Referências

[ABNT 2011] ABNT (2011). Digital terrestrial television - data coding and transmission specification for digital broadcasting - part 2: Ginga-ncl for fixed and mobile receivers - xml application language for application coding. ABNT NBR 15606-2:2011 standard.

[Adobe 2010] Adobe (2010). *ActionScript references and documentation*. Adobe Systems Incorporated.

[Allen 1983] Allen, J. F. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843.

[Araújo et al. 2008] Araújo, E. C., Azevedo, R. G. A., and Neto, C. S. S. (2008). Ncl-validator: um processo para validação sintática e semântica de documentos multimídia ncl. In *Jornada de Informática do Maranhão*. in portuguese.

[ARIB 2014] ARIB (2014). Data coding and transmission specification for digital broadcasting. Standard B24 v6.1.

[ATSC 2009] ATSC (2009). Advanced common application platform (acap). Document A/101A.

[Barreto et al. 2016] Barreto, F., Tamaki, D., dos Santos, J. A. F., and Muchaluat-Saade, D. C. (2016). Ncl-tester: Ferramenta gráfica para criação de testes temporais para documentos ncl. In *Proceedings of the 22th Brazilian Symposium on Multimedia and the Web*, WebMedia '16. ACM.

[Bertino et al. 2005] Bertino, E., Ferrari, E., Perego, A., and Santi, D. (2005). A constraint-based approach for the authoring of multi-topic multimedia presentations. In *IEEE International Conference on Multimedia and Expo*, pages 578–581, Amsterdam, Netherlands. IEEE Computer Society.

[Blakowski and Steinmetz 1996] Blakowski, G. and Steinmetz, R. (1996). A media synchronization survey: Reference model, specification and case studies. *Journal on Selected Areas in Communications*, 14(1):5–35.

[Boll 2001] Boll, S. (2001). *ZYX - Towards flexible multimedia document models for reuse and adaptation*. PhD thesis, Vienna University of Technology.

[Bouyakoub and Belkhir 2008] Bouyakoub, S. and Belkhir, A. (2008). H-smil-net: A hierarchical petri net model for smil documents. In *Proceedings of the Tenth International Conference on Computer Modeling and Simulation*, UKSIM '08, pages 106–111, Washington, DC, USA. IEEE Computer Society.

[Bouyakoub and Belkhir 2011] Bouyakoub, S. and Belkhir, A. (2011). Smil builder: An incremental authoring tool for smil documents. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)*, 7(1):2:1–2:30.

[Buchanan and Zellweger 1992] Buchanan, M. C. and Zellweger, P. T. (1992). Specifying temporal behavior in hypermedia documents. In *Proceedings of the ACM conference on Hypertext*, pages 262–271. ACM.

[Buchanan and Zellweger 2005] Buchanan, M. C. and Zellweger, P. T. (2005). Automatic temporal layout mechanisms revisited. *ACM Transactions on Multimedia Computing, Communications and Applications*, 1(1):60–88.

[Bulterman et al. 2013] Bulterman, D., Cesar, P., and Guimaraes, R. L. (2013). Socially-aware multimedia authoring: Past, present, and future. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)*, 9(1s):35.

[Damasceno et al. 2014] Damasceno, J. R., dos Santos, J. A. F., and Muchaluat-Saade, D. C. (2014). Editec - a graphical editor for hypermedia composite templates. *Multimedia Tools and Applications*, 70(2):1167–1198.

[De Moura and Bjørner 2011] De Moura, L. and Bjørner, N. (2011). Satisfiability modulo theories: Introduction and applications. *Communications of the ACM*, 54(9):69–77.

[de Oliveira et al. 2001] de Oliveira, M., Turine, M., and Masiero, P. (2001). A statechart-based model for hypermedia applications. *ACM Transactions on Information Systems (TOIS)*, 19(1):52.

[Deltour and Roisin 2006] Deltour, R. and Roisin, C. (2006). The limsee3 multimedia authoring model. In *Proceedings of the 2006 ACM symposium on Document engineering*, pages 173–175. ACM.

[Díaz et al. 2001] Díaz, P., Aedo, I., and Panetsos, F. (2001). Modeling the dynamic behavior of hypermedia applications. *IEEE Transactions on Software Engineering*, 27(6):550–572.

[dos Santos 2012] dos Santos, J. A. F. (2012). Multimedia and hypermedia document validation and verification using a model-driven approach. Master's thesis, UFF.

[dos Santos 2016] dos Santos, J. A. F. (2016). *Multimedia Document Validation Along its Life Cycle*. PhD thesis, Universidade Federal Fluminense.

[dos Santos et al. 2013] dos Santos, J. A. F., Braga, C., and Muchaluat-Saade, D. C. (2013). Automating the analysis of ncl documents with a model-driven approach. In *Proceedings of the 19th Brazilian Symposium on Multimedia and the Web*, WebMedia '13, pages 193–200, New York, NY, USA. ACM.

[dos Santos et al. 2015] dos Santos, J. A. F., Braga, C., and Muchaluat-Saade, D. C. (2015). A rewriting logic semantics for ncl. *Science of Computer Programming*, 107–108:64 – 92.

[dos Santos and Muchaluat-Saade 2012] dos Santos, J. A. F. and Muchaluat-Saade, D. C. (2012). Xtemplate 3.0: spatio-temporal semantics and structure reuse for hypermedia compositions. *Multimedia Tools and Applications*, 61(3):645–673.

[Elias et al. 2006] Elias, S., Easwarakumar, K., and Chbeir, R. (2006). Dynamic consistency checking for temporal and spatial relations in multimedia presentations. In *Proceedings of the 2006 ACM symposium on Applied computing*, pages 1380–1384, Dijon, France. ACM.

[ETSI 2008] ETSI (2008). Digital video broadcasting (dvb); multimedia home platform (mhp) specification 1.0.3. ETSI ES 201 812 v1.1.2.

[Felix 2004] Felix, M. F. (2004). *Formal Analysis of Software Models Oriented by Architectural Abstractions*. PhD thesis, Pontifícia Universidade Católica do Rio de Janeiro. in Portuguese.

[Furuta and Stotts 2001] Furuta, R. and Stotts, P. D. (2001). Trellis: a formally-defined hypertextual basis for integrating task and information. *Coordination Theory and Collaboration Technology*, pages 341–367.

[Gaggi and Bossi 2011] Gaggi, O. and Bossi, A. (2011). Analysis and verification of smil documents. *Multimedia Systems*, 17(6):487–506.

[Ghinea et al. 2014] Ghinea, G., Timmerer, C., Lin, W., and Gulliver, S. R. (2014). Mulsemedia: State of the art, perspectives, and challenges. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 11(1s):17.

[Harel 1987] Harel, D. (1987). Statecharts: A visual formalism for complex systems. *Science of computer programming*, 8(3):231–274.

[ISO/IEC 2005] ISO/IEC (2005). Information technology - coding of audio-visual objects - part 11: Scene description and application engine. ISO/IEC 14496-11:2005.

[ITU 2009] ITU (2009). Nested context language (ncl) and ginga-ncl for iptv services. http://www.itu.int/rec/T-REC-H.761-200904-S. ITU-T Recommendation H.761.

[Jourdan et al. 1998] Jourdan, M., Layaida, N., Roisin, C., Sabry-Ismail, L., and Tardif, L. (1998). Madeus, an authoring environment for interactive multimedia documents. In *Proceedings of the 6th ACM International Conference on Multimedia*, pages 267–272. ACM.

[Júnior et al. 2012] Júnior, D. P., Farines, J.-M., and Koliver, C. (2012). An approach to verify live ncl applications. In *Proceedings of the 18th Brazilian Symposium on Multimedia and the Web*, WebMedia '12, pages 223–232, New York, NY, USA. ACM.

[King et al. 2004] King, P., Schmitz, P., and Thompson, S. (2004). Behavioral reactivity and real time programming in xml: functional programming meets smil animation. In *Proceedings of the 2004 ACM symposium on Document engineering*, pages 57–66. ACM.

[Laborie et al. 2011] Laborie, S., Euzenat, J., and Layaïda, N. (2011). Semantic adaptation of multimedia documents. *Multimedia tools and applications*, 55(3):379–398.

[Lemlouma and Layaïda 2004] Lemlouma, T. and Layaïda, N. (2004). Context-aware adaptation for mobile devices. In *IEEE International Conference on Mobile Data Management*, pages 106–111.

[Muchaluat-Saade 2003] Muchaluat-Saade, D. C. (2003). *Relations in Hypermedia Authoring Languages: Improving Reuse and Expressiveness*. PhD thesis, Pontifícia Universidade Católica do Rio de Janeiro.

[Na and Furuta 2001] Na, J. and Furuta, R. (2001). Dynamic documents: authoring, browsing, and analysis using a high-level petri net-based hypermedia system. In *Proceedings of the 2001 ACM Symposium on Document engineering*, pages 38–47. ACM.

[Neto et al. 2012] Neto, C. d. S. S., Soares, L. F. G., and de Souza, C. S. (2012). Tal-template authoring language. *Journal of the Brazilian Computer Society*, 18(3):185–199.

[Neto et al. 2011] Neto, J. R. C., Santos, R. C. M., Neto, C. S. S., and Teixeira, M. M. (2011). Método de validação estrutural e contextual de documentos ncl. In *Proceedings of the 17th Brazilian Symposium on Multimedia and the Web*, pages 1–8. in Portuguese.

[Peterson 1981] Peterson, J. L. (1981). *Petri Net Theory and Modeling of systems*. Prentice-Hall.

[Santos et al. 1998] Santos, C., Soares, L., de Souza, G., and Courtiat, J. (1998). Design methodology and formal validation of hypermedia documents. In *Proceedings of the sixth ACM International Conference on Multimedia*, pages 39–48, Bristol, United Kingdom. ACM.

[Sarkis et al. 2014] Sarkis, M., Concolato, C., and Dufourd, J.-C. (2014). The virtual splitter: Refactoring web applications for themultiscreen environment. In *Proceedings of the 2014 ACM Symposium on Document Engineering*, DocEng '14, pages 139–142, New York, NY, USA. ACM.

[Soares et al. 2012] Soares, L. F. G., Neto, C. S. S., and Sousa, J. G. (2012). Architecture for hypermedia dynamic applications with content and behavior constraints. In *ACM symposium on Document engineering*, pages 217–226. ACM.

[Soares and Rodrigues 2005] Soares, L. F. G. and Rodrigues, R. F. (2005). Nested context model 3.0 part 1 - ncm core. Technical report, Informatics Department, PUC-Rio, Rio de Janeiro.

[Soares et al. 2006] Soares, L. F. G., Rodrigues, R. F., Costa, R. R., and Moreno, M. F. (2006). Nested context language 3.0 part 9 - ncl live editing commands. Technical report, Informatics Department, PUC-Rio, Rio de Janeiro.

[Soares et al. 2000] Soares, L. F. G., Rodrigues, R. F., and Muchaluat-Saade, D. C. (2000). Modeling, authoring and formatting hypermedia documents in the hyperprop system. *Multimedia Systems*, 8(2):118–134.

[Sun et al. 1998] Sun, C., Jia, X., Zhang, Y., Yang, Y., and Chen, D. (1998). Achieving convergence, causality preservation, and intention preservation in real-time cooperative editing systems. *ACM Trans. Comput.-Hum. Interact.*, 5(1):63–108.

[Teixeira et al. 2012] Teixeira, C. A. C., Melo, E. L., Freitas, G. B., Santos, C. A. S., and Pimentel, M. d. G. C. (2012). Discrimination of media moments and media intervals: sticker-based watch-and-comment annotation. *Multimedia Tools and Applications*, 61(3):675–696.

[W3C 1999] W3C (1999). Synchronized multimedia integration language (smil) document object model. http://www.w3.org/TR/SMIL/smil-DOM.html. World-Wide Web Consortium Recommendation.

[W3C 2000] W3C (2000). Document object model (dom) level 2 core specification. World-Wide Web Consortium Recommendation DOM-Level-2-Core-20001113.

[W3C 2008a] W3C (2008a). Extensible markup language (xml) 1.0 (fifth edition). World-Wide Web Consortium Recommendation.

[W3C 2008b] W3C (2008b). Synchronized multimedia integration language - smil 3.0 specification. http://www.w3c.org/TR/SMIL3. World-Wide Web Consortium Recommendation.

[W3C 2014] W3C (2014). Html5: A vocabulary and associated apis for html and xhtml. World-Wide Web Consortium Candidate Recommendation.

[W3C 2014] W3C (2014). Web animations 1.0. http://www.w3.org/TR/web-animations/. World-Wide Web Consortium Working Draft.

[Wahl and Rothermel 1994] Wahl, T. and Rothermel, K. (1994). Representing time in multimedia systems. In *Proceedings of International Conference on Multimedia Computing and Systems*. IEEE Computer Society Press.

[Willrich et al. 2001] Willrich, R., de Saqui-Sannes, P., Sénac, P., and Diaz, M. (2001). Design and management of multimedia information systems. pages 380–411.

Biografia Resumida dos Autores

Dr. Joel dos Santos possui graduação em engenharia de telecomunicações (2009), mestre (2012) e doutor (2016) em Ciência da Computação pela Universidade Federal Fluminense (UFF). É professor da Escola de Informática e Computação (EIC) do Centro Federal de Educação Tecnológica Celso Suckow da Fonseca (CEFET/RJ), atuando em cursos relacionados à Ciência da Computação nos níveis Técnico, Graduação e Pós-Graduacão. Atua desde 2006 em áreas relacionadas a multimídia, dentre as quais destacam-se: Autoria multimídia, Multissesorial mídia e TV digital. Pelo laboratório MídiaCom (UFF) participou do desenvolvimento da Linguagem XTemPlate 3.0 e das APIs aNa e aNaa. Durante intercâmbio na Universität Ulm, Alemanha, tive a oportunidade de trabalhar em projeto de pesquisa no OMI (*Institut für Organization und Management von Informationssytemen*). Como parte do programa Ciência sem Fronteiras trabalhei como pesquisador convidado do grupo Tyrex no Inria (*Institut National de Recherche en Informatique et en Automatique*) em Grenoble, França.

Dr. Débora Christina Muchaluat Saade possui graduação em Engenharia de Computação (1992), mestrado em Informática (1996) e doutorado em Informática pela Pontifícia Universidade Católica do Rio de Janeiro (2003). Desde 2002, é professora da Universidade Federal Fluminense. Integrou o corpo docente do Departamento de Engenharia de Telecomunicações até maio de 2009 e desde então faz parte do corpo docente do Instituto de Computação. É bolsista de produtividade DT do CNPq e foi jovem cientista pela FAPERJ. Tem experiência nas áreas de Ciência da Computação e Engenharia de Telecomunicações, com ênfase em Teleinformática, atuando principalmente nos seguintes temas: redes de computadores, redes sem fio, sistemas multimídia e hipermídia distribuídos, linguagens de autoria hipermídia, televisão digital interativa e telemedicina. Participou do desenvolvimento da linguagem NCL - Nested Context Language - adotada como padrão ABNT NBR 15606-2 no middleware GINGA do Sistema Brasileiro de TV Digital e como recomendação internacional do ITU-T H.761 para serviços IPTV.