TV Digital Interativa

Conceitos, Tecnologias e Aplicações

Roteiro

1. O que é a TV Digital Interativa (TVDI) ?
2. Modelo de TVDI;
3. Tipos de Definição na TVD;
4. Aplicações na TVDI;
5. Middleware;
6. Padrões utilizados;
7. Set-Top Box (STB);
8. JavaTV;
9. NCL;
10. Links sobre TVDI.

1. O que é a TVDI ?

- O sinal da TV é transmitido em um formato digital, que permite multiplexar informações de áudio,vídeo e dados num mesmo canal de transmissão;
- Uso de um canal de retorno para a comunicação com o emissor do sinal, o que caracteriza a interatividade.

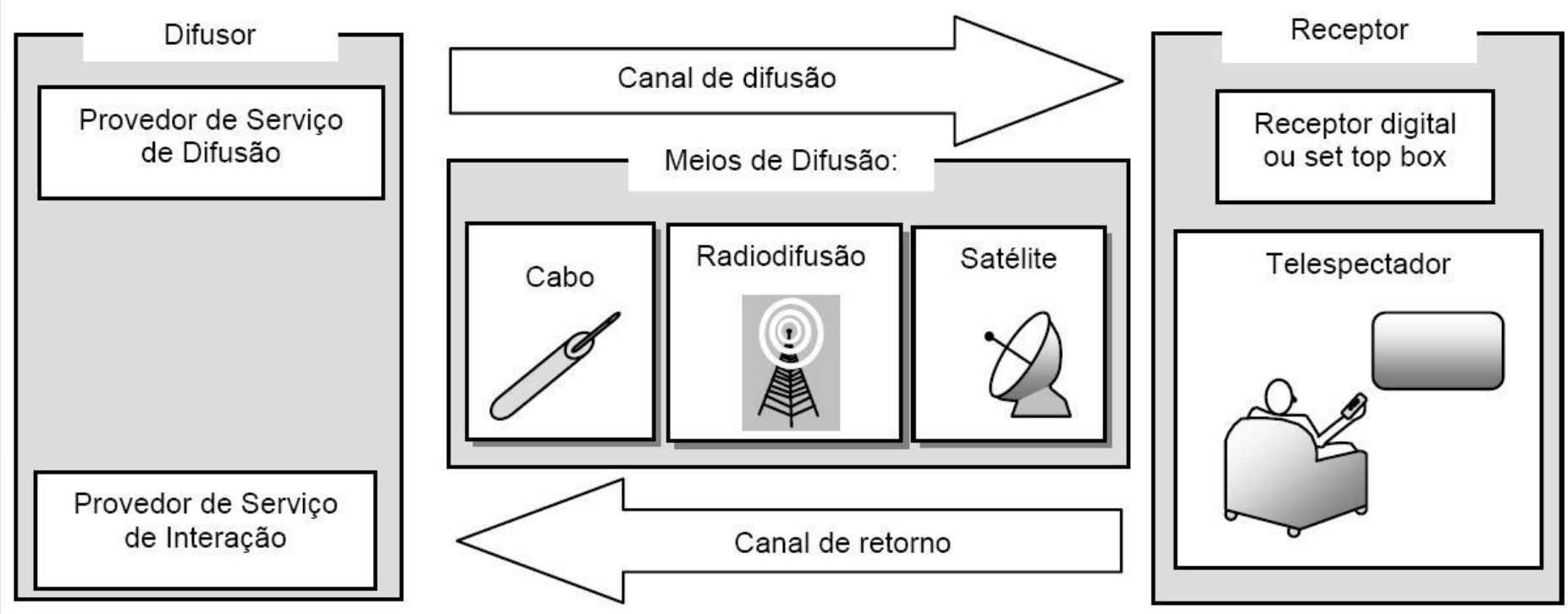
1. O que é a TVDI ?

- Como funciona ?
  - A TV Digital funciona convertendo em fluxos de bits, imagens e sons (e dados), que por sua vez são transformados em sinas elétricos, que são transportados por ondas magnéticas.
  - Então os sinais elétricos chegam ao receptor, e são novamente transformados em fluxos de bits e convertidos em imagens e sons (e dados).

1. O que é a TVDI ?

- A verdadeira TV Digital, é completamente digital e envolve:
  - **câmeras digitais** funcionando com uma resolução muito maior do que as câmeras analógicas;
  - **transmissão digital;**
  - **exibição digital** com uma resolução muito maior.

2. Modelo de TVDI



3. Tipos de Definição da TVD

- Standard Definition TV

  640 pontos por 480 linhas – 4:3

- High Definition TV

  1280 pontos por 720 linhas – 16:9

- Full High Definition TV

  1920 pontos por 1080 linhas – 16:9

4. Aplicações na TVDI

Segundo o Emarketer, empresa americana de pesquisa em novas tecnologias, o termo TVDI abrange uma série de aplicações, serviços e tecnologias, muitas ainda nem inventadas.

Mas é possível classificar toda variedade de informações abrangidas pelo termo em 7 grandes grupos:

4. Aplicações na TVDI

1. *Enhanced TV*: tipo de conteúdo televisivo que abrange texto, vídeo e elementos gráficos, como fotos e animações;
2. *Internet on TV*: permite o acesso à internet usando o televisor. Todas as funções da internet que conhecemos hoje estão disponíveis;

4. Aplicações na TVDI

3. *Individualized TV*: permite a adaptação total da TV ao gosto do telespectador;
4. *Video-on-demand*: capacita os telespectadores a assistir o programa na hora em que quiserem, sem a restrição ao horário em que é transmitido pela emissora;

4. Aplicações na TVDI

5. *Personal Video Recorder (PVR)*: também conhecido como *Personal TV ou Digital Video Recorder (DVR)*, permite a gravação digital de programas apenas especificando o título, o horário, o assunto, o ator, ou algum outro dado pré-cadastrado sobre o filme;

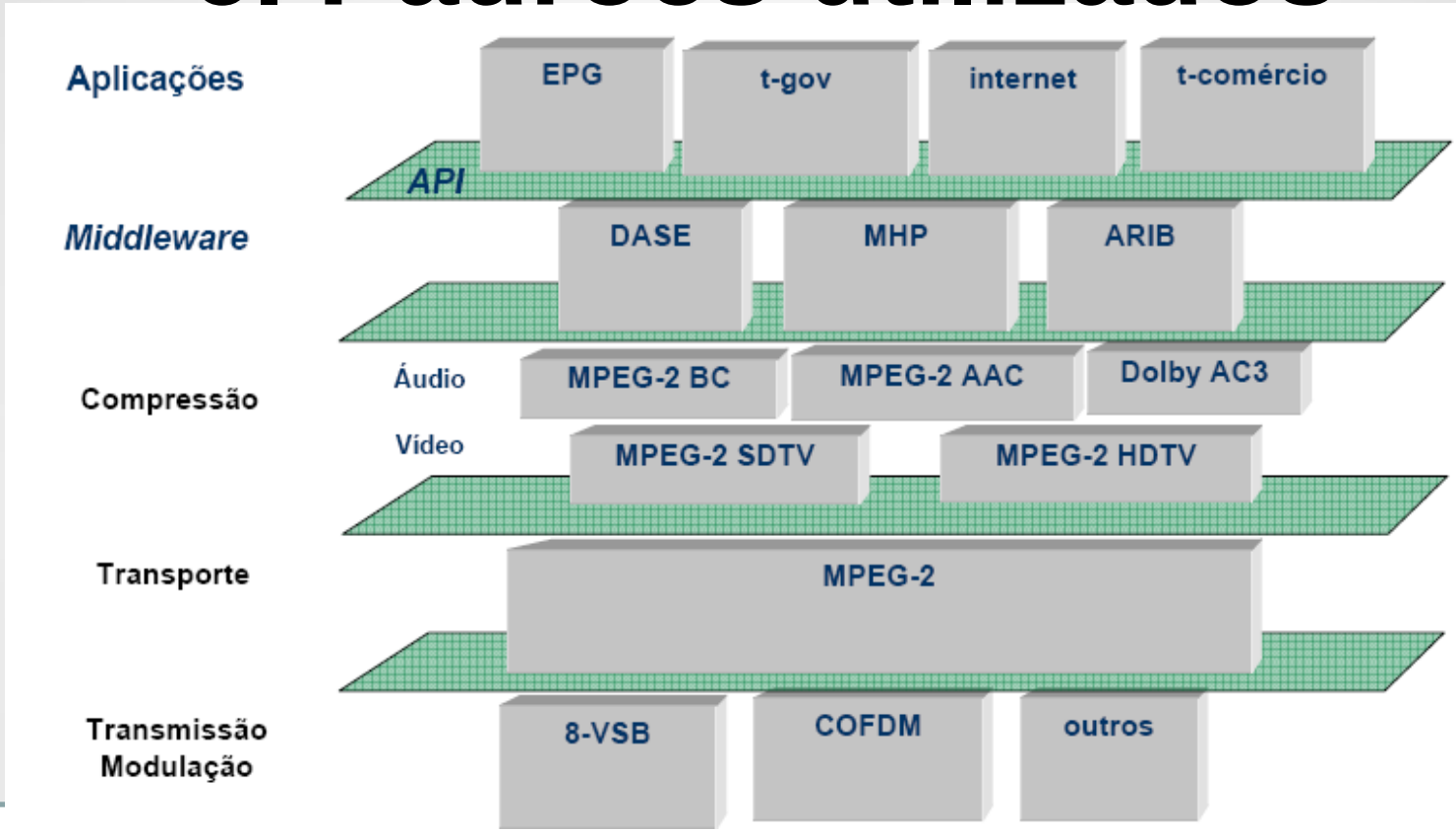
4. Aplicações na TVDI

6. *Walled Garden*: um portal contendo um guia das aplicações interativas. Pode ser comparado à revista com a grade de programação das TVs a cabo;
7. *Game Console*: permite o uso da TV para jogos, seja contra a própria TV, computador, ou em rede, contra outros jogadores.

5. Middleware

- Esconde as peculiaridades e complexidades do *hardware*, sistema operacional, *drivers* de dispositivos, *software* e *hardware* responsáveis pela decodificação do sinal.
- O *middleware* é a camada de comunicação entre a camada de aplicação e os serviços oferecidos pelas camadas inferiores.

6. Padrões utilizados



	Europeu (DVB)	**Americano (ATSC)**	**Japonês (ISDB)**	**Brasil (ISDB)**
Middleware	MHP	DASE	ARIB	Ginga
Comp.Áudio	MPEG-2 BC	Dolby AC3	MPEG-2 AAC	MPEG-4
Comp.Vídeo	MPEG-2 SDTV	MPEG-2 HDTV	MPEG-2 HDTV	MPEG-4
Transporte	MPEG-2	MPEG-2	MPEG-2	MPEG-2
Transmissão	COFDM	8-VSB.	COFDM	COFDM

7. Set-Top Box (STB)

• “Dispositivo de Entretenimento Interativo residencial que disponibiliza acesso à internet, streaming de vídeo e informações via tradicional sistema de televisão”, National Semiconductor - 2000.

- Conversão do sinal digital para formato analógico;
- Seleção de programação, inibição de comerciais, gravação de filmes etc.
- Download e Upload de dados;
- Jogos;
- Ensino a distância;
- Etc.

8. JavaDTV

- É uma plataforma para desenvolvimento e distribuição de serviços para TVD;
- O middleware dos 3 (e o brasileiro) principais sistemas de TVD adotam Java;
- Paradigma PROCEDURAL;
- http://java.sun.com/

9. NCL

- Nested Context Language (PUC-Rio), baseada no NCM (Nested Context Model);
- Paradigma DECLARATIVO;
- http://www.ncl.org.br/
- O middleware brasileiro GINGA está sendo construído para executar aplicações desenvolvidas em Java e NCL.

10. Links sobre TVDI

- http://uitv.info/
- http://www.labitv.futuro.usp.br/
- http://www.interactivetvweb.org/
- http://www.itvt.com/
- http://www.pjb.co.uk/
- http://www.lsda.org.uk/
- http://www.digitaltelevision.gov.uk/
- http://www.broadbandbananas.com/
- http://sbtvd.cpqd.com.br/
- http://www.forumsbtvd.org.br/
- http://www.ncl.org.br
- http://www.ginga.org.br
- http://www.tiresias.org/guidelines/television/index.htm

Nested Context Language

(Linguagem de Contexto Aninhado)

Nested Context Language
(Linguagem de Contexto Aninhado)

A linguagem NCL Nested Context Language é uma linguagem declarativa para autoria de documentos ***hipermídia*** baseados no modelo conceitual NCM Nested Context Model - que foi desenvolvida utilizando uma estrutura modular, seguindo os princípios adotados pelo W3C.

Fonte: **www.ncl.org.br**

Nested Context Language

- Linguagem proposta pela PUC – Rio;
- Paradigma Declarativo
  - ex: NCL, HTML, SQL
  - **“o que fazer”**
  - linguagens de domínio específico

- É especificado por XML Schema;
  - Defini a estrutura do documento de forma auto-descritiva;

- A linguagem é baseada no modelo de contexto aninhado (NCM);
  - Permite que documentos NCL possuam mais de um nível de aninhamento de contexto;

Nested Context Language

Exemplo de XML

```
<livro>
        <capitulo>
           <secao>
                <paragrafo>...</paragrafo>
                <paragrafo>...</paragrafo>
            </secao>
            <secao>
                <paragrafo>...</paragrafo>
                <paragrafo>...</paragrafo>
            </secao>
        </capitulo>
</livro>
```

- Todas as tags devem ser fechadas;
- São case sensitive;
- Devem estar bem aninhadas;
- Deve possuir uma raiz;
- Tags principais do NCL:

  <ncl> (Raiz)

  <head>

  <body>

Nested Context Language

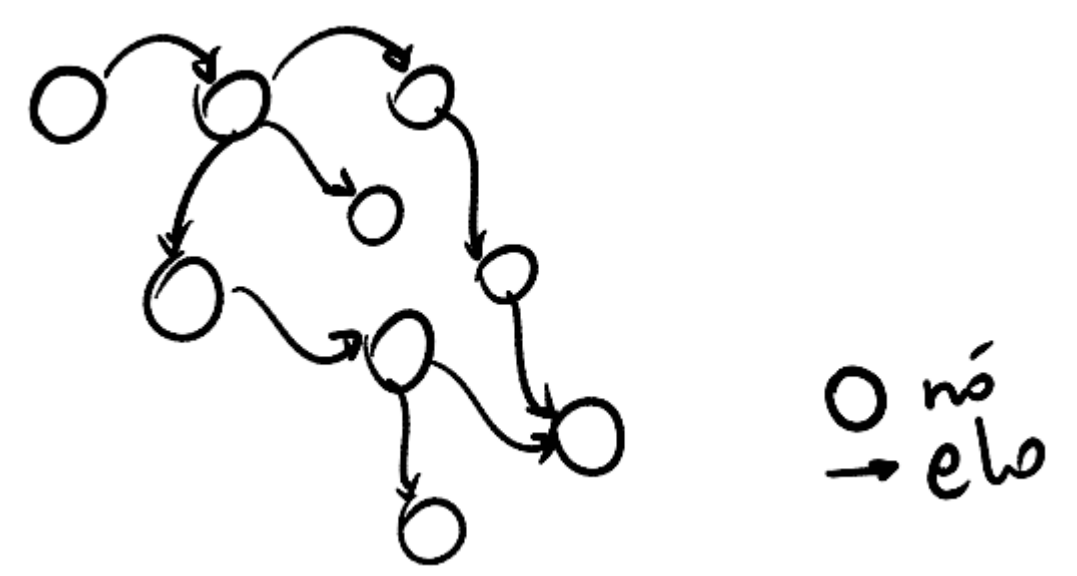
- Projetada de forma modular:
  - Permitindo a combinação de seus módulos em diferentes perfis de linguagem;
- Principais Módulos:
  - Structure > atributos que devem estar em todos os documentos
  - Layout > como as mídias serão inicialmente apresentadas
  - Media >representar conteúdo físico de mídia digital
  - Context > definir contextos internos, como links nas mídias
  - Descriptor > informação necessária para que a mídia seja corretamente exibida
  - Linking > possibilita a relação entre elementos hipermídia
  - Connector > estabelece eventos de sincronização e interação com o conteúdo

Nested Context Language

- Linguagem de marcação de dados:
  - Formato para descrever dados estruturados;
  - Declaração mais precisa dos dados;
- Suporte a sincronização;
  - Baseada na estrutura;
  - Suporte a canal de retorno;

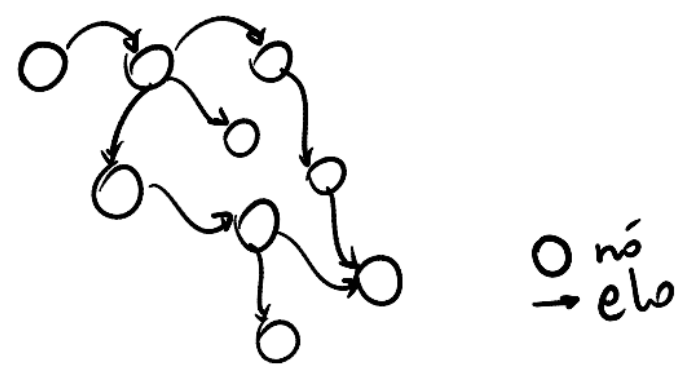
- Suporte a múltiplos dispositivos;
- Suporte a edição ao vivo;
  - TV Social;
- NCL é software livre!

Nested Context Language

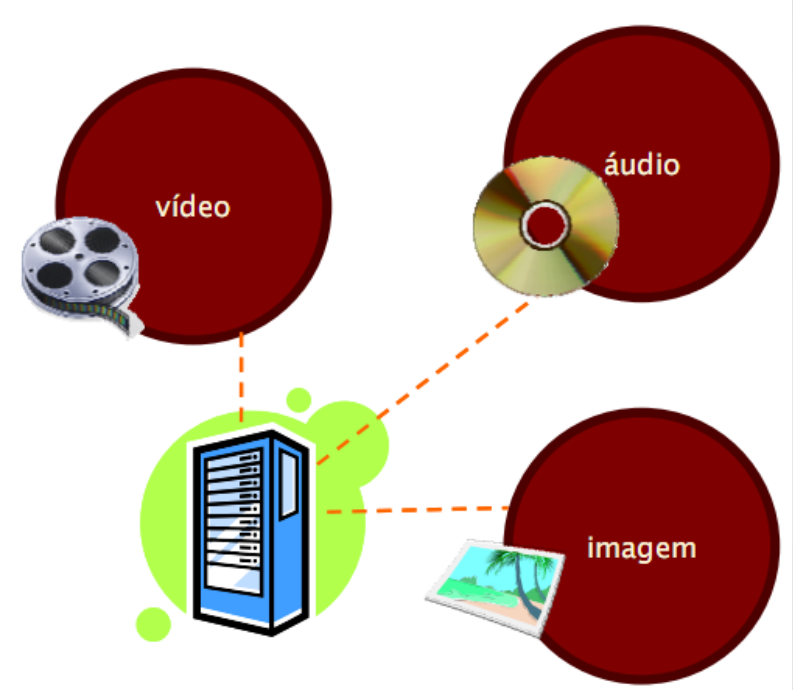


Nós e elos num documento hipermídia

Nested Context Language

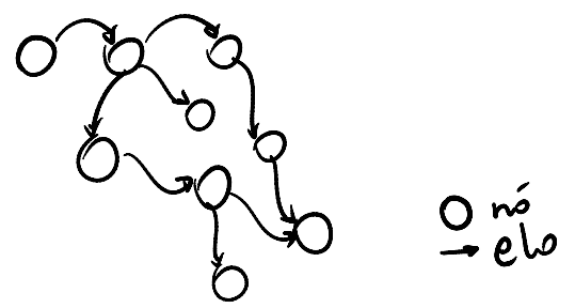


Nós e elos num documento hipermídia

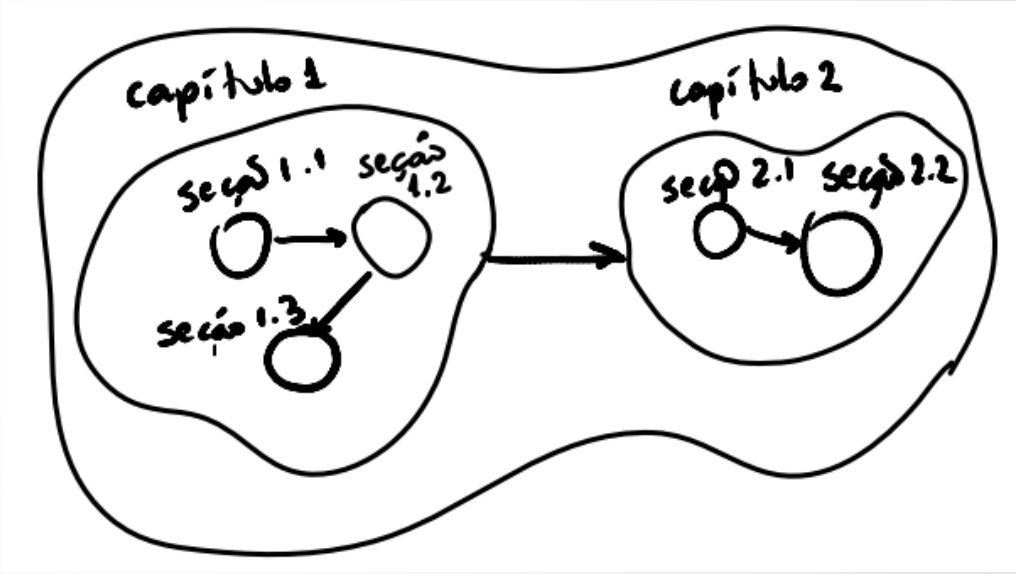


Nós de mídia

Nested Context Language



Nós e elos num documento hipermídia



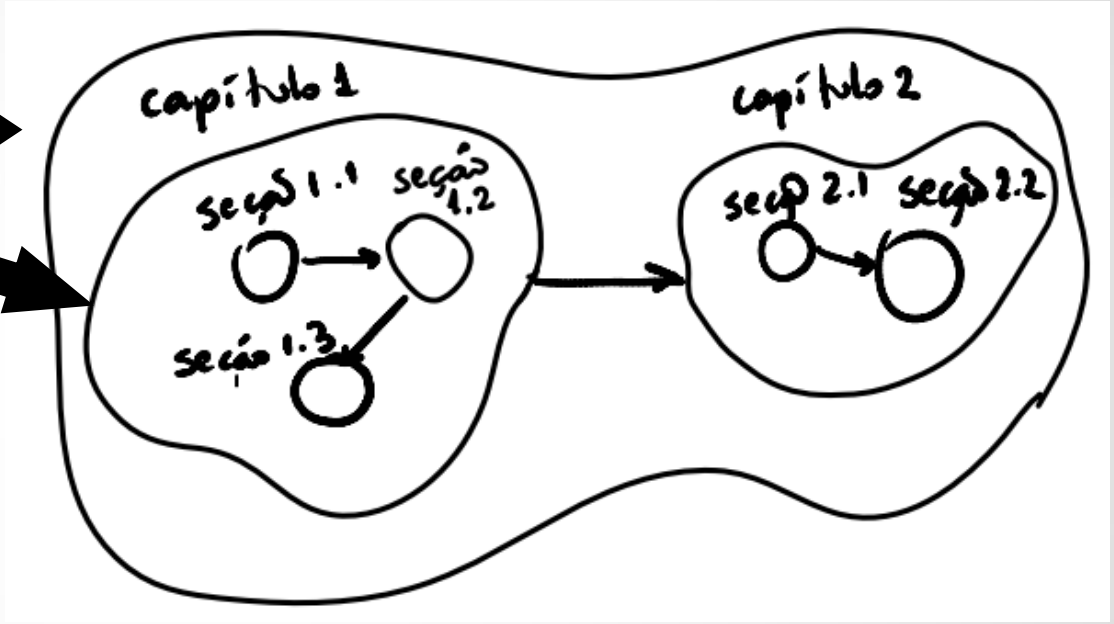
Nós, elos e nós de composição (contextos)

Nested Context Language

Nós de Contexto ou Composição

Conjunto de nós ou conjunto de composições

Daí o nome de modelo **Contextos Aninhados**.



Nós, elos e nós de composição (contextos)

Nested Context Language
(Linguagem de Contexto Aninhado)

A linguagem NCL Nested Context Language é uma linguagem declarativa para ***autoria de documentos hipermídia*** baseados no modelo conceitual NCM Nested Context Model - que foi desenvolvida utilizando uma estrutura modular, seguindo os princípios adotados pelo W3C.

Fonte: **www.ncl.org.br**

Multimídia

É a combinação, controlada por computador, de pelo menos um tipo de mídia estática (texto, fotografia, gráfico), com pelo menos um tipo de media dinâmica (vídeo, aúdio, animação) (Chapman & Chapman 2000 e Fluckiger 1995).

ginga NCL Lua

LIMS
Laboratory of Innovation on Multimedia Systems

Multimídia = ***Hipermídia***?

Multimídia X Hipermídia

Theodor Holm Nelson

Inventou os termos **hipertexto** e **hipermídia** em 1963

Multimídia X Hipermídia

Theodor Holm Nelson

Hipermídia = A utilização do hipertexto em aplicações capazes de integrar não só texto mas também imagem e som.

Hipermídia = Multimídia Interativa

Multimídia X Hipermídia

Sistema Multímidia

Sistema capaz de manipular ao menos um tipo de mídia discreta e um tipo de mídia continua na forma digital de maneira sincronizada

Sistema Hipermídia

Sistema Multimídia capaz de ***tratar eventos*** causados pela interação com o usuário e ***reagir a esses eventos***

Nested Context Language
(Linguagem de Contexto Aninhado)

A linguagem NCL Nested Context Language é uma linguagem declarativa para autoria de documentos ***hipermídia baseados no modelo conceitual NCM Nested Context Model*** - que foi desenvolvida utilizando uma estrutura modular, seguindo os princípios adotados pelo W3C.

Fonte: **www.ncl.org.br**

NCM – Nested Context Model

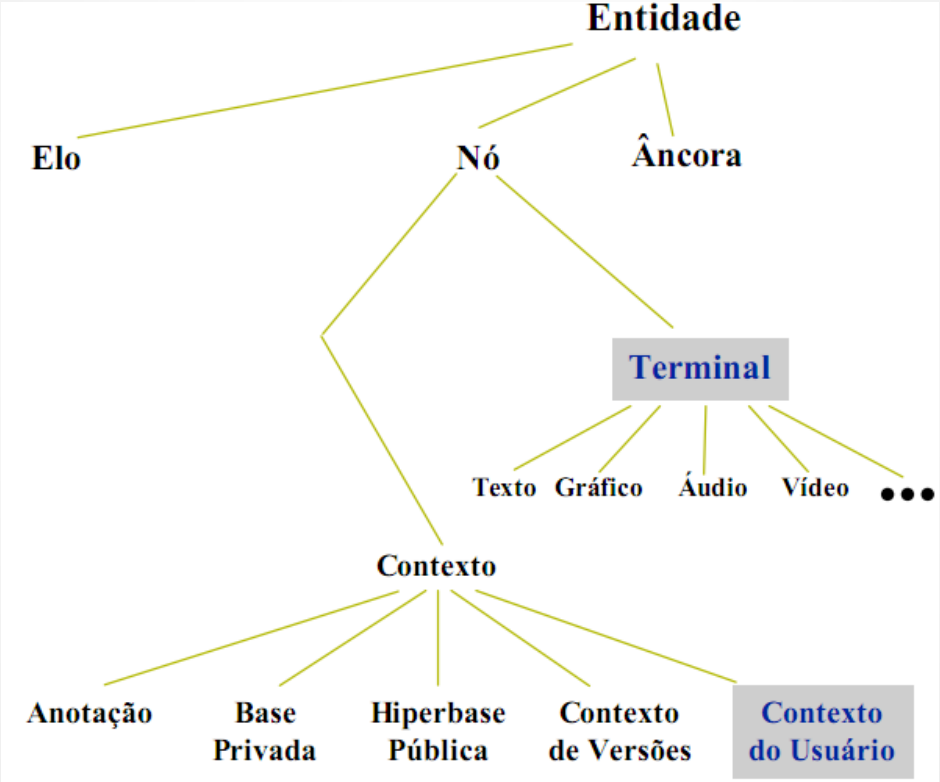
(Modelo de Contexto Aninhado)

- É um modelo conceitual centrado na representação e tratamento de documentos hipermídia e que dá suporte ao NCL;
- Cujo modelo de interface separa os componentes de dados e de exibição de objetos;

NCM – Nested Context Model

(Modelo de Contexto Aninhado)

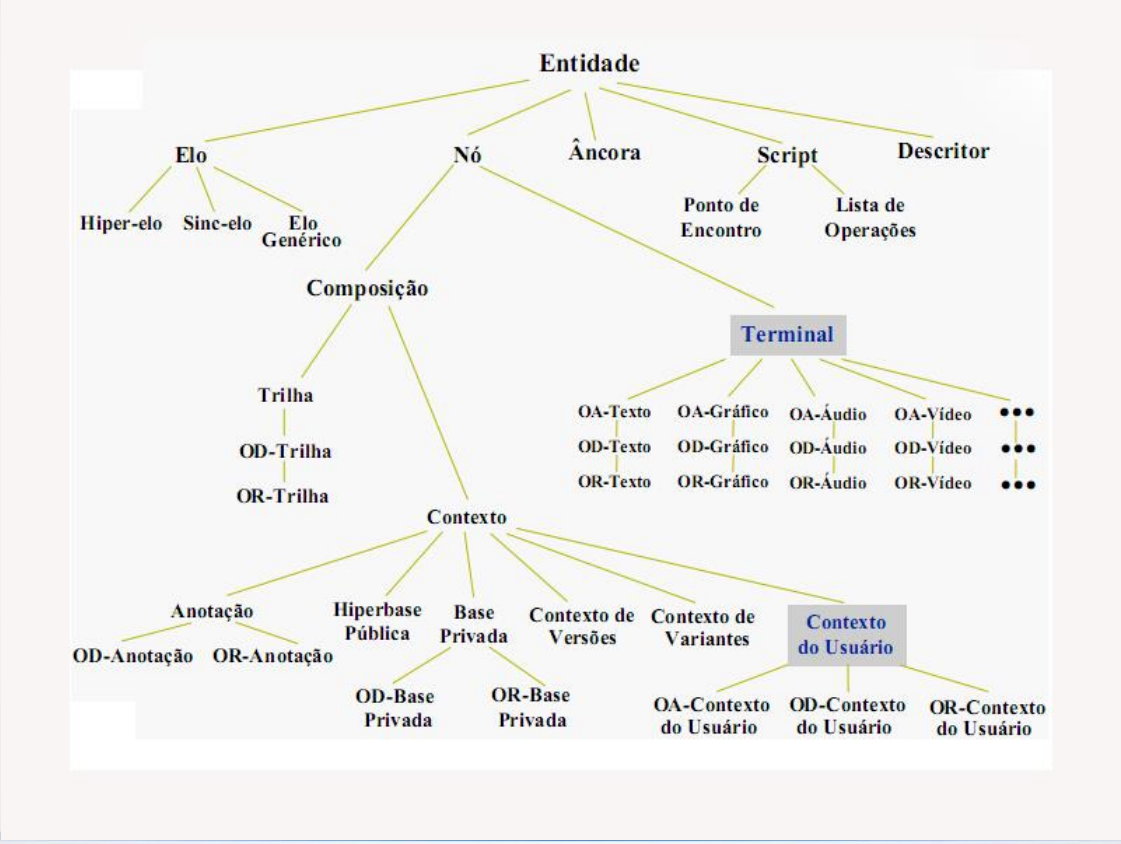
Hierarquia de Classes NCM (Versão Anterior a 2.2)



NCM – Nested Context Model

(Modelo de Contexto Aninhado)

Hierarquia de Classes NCM (Versão 2.2)



NCM – Nested Context Model

(Modelo de Contexto Aninhado)

- É um modelo conceitual centrado na representação e tratamento de documentos hipermídia;

- A partir da versão 2.2 foi imposto o conceito de evento;
  - Com a definição de eventos possibilitou a realização de sincronização espacial e temporal entre nós.

NCM – Nested Context Model

(Modelo de Contexto Aninhado)

María José Peréz-Luque

"um evento é uma ocorrência no tempo que pode ser instantânea ou durar um período de tempo"

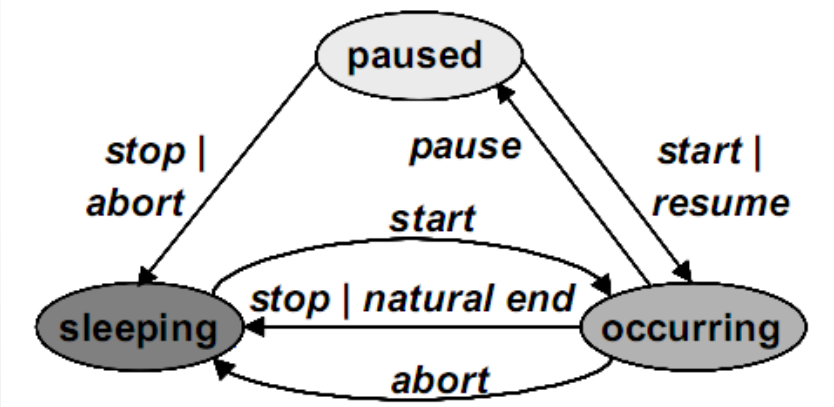
NCM – Nested Context Model

(Modelo de Contexto Aninhado)

Para o NCM:

Um evento é a exibição.

Um evento NCM pode estar em um dos seguintes estados: dormindo (sleeping), ocorrendo (occurring) ou suspendo (paused);

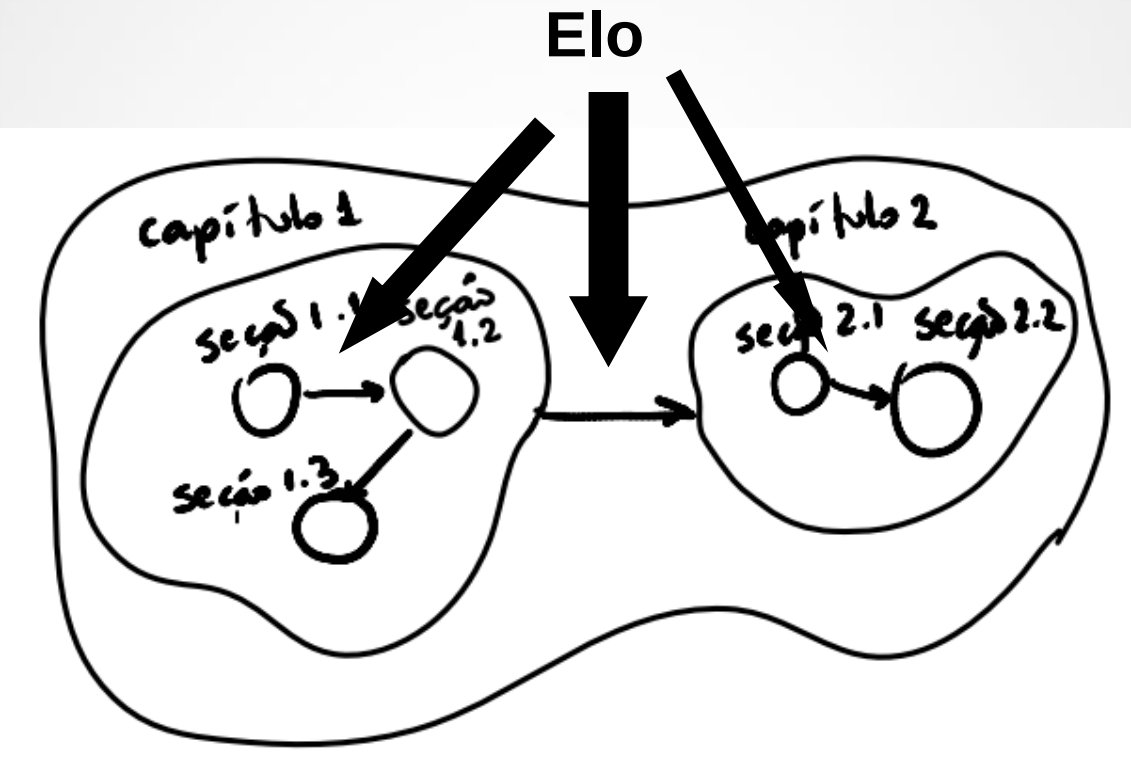


NCM – Nested Context Model

(Modelo de Contexto Aninhado)

- Com a introdução deste conceito permitiu:
  - A criação do Descritor;
  - Elos redefinidos permitindo relações de sincronização temporal e espacial entre os nós;

NCM 3.0



NCM 3.0

Elo

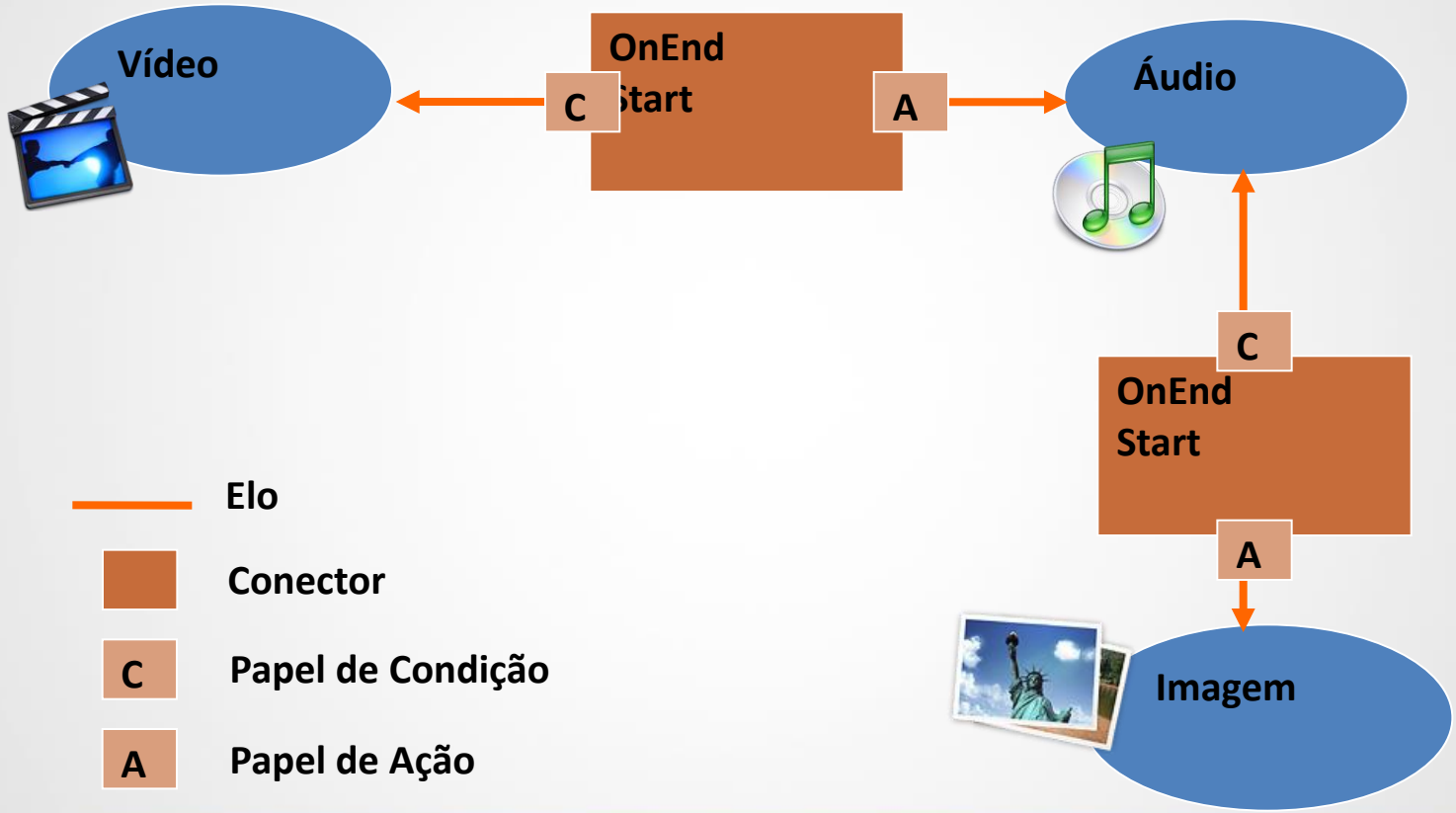
- É uma entidade NCM que possui duas propriedades adicionais: um conector e um conjunto de associações a esse conector;
- Criação do Conector na versão 3.0;
  - Defini condições entre as relações hipermídia, ou entre os nós.

NCM 3.0

Conector

- Bind
  - Define os participantes;
- Papel
  - A sua definição é baseada no conceito de eventos;
  - Define a função dos participantes na relação;

NCM 3.0


Vídeo
C
OnEnd
Start
A
Áudio
C
OnEnd
Start
A
Imagem
Elo
Conector
C
Papel de Condição
A
Papel de Ação

NCL

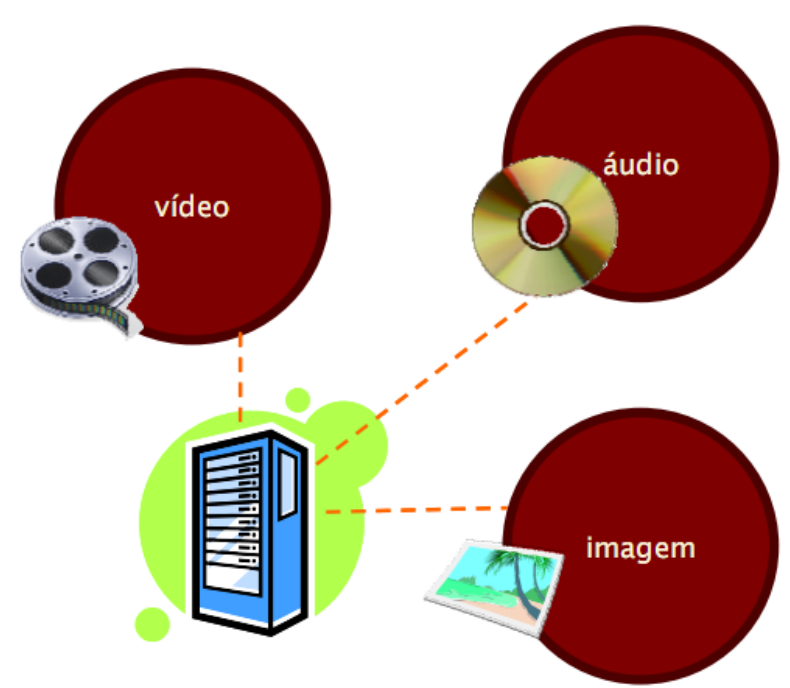
O quê?

Onde?

Como?

Quando?

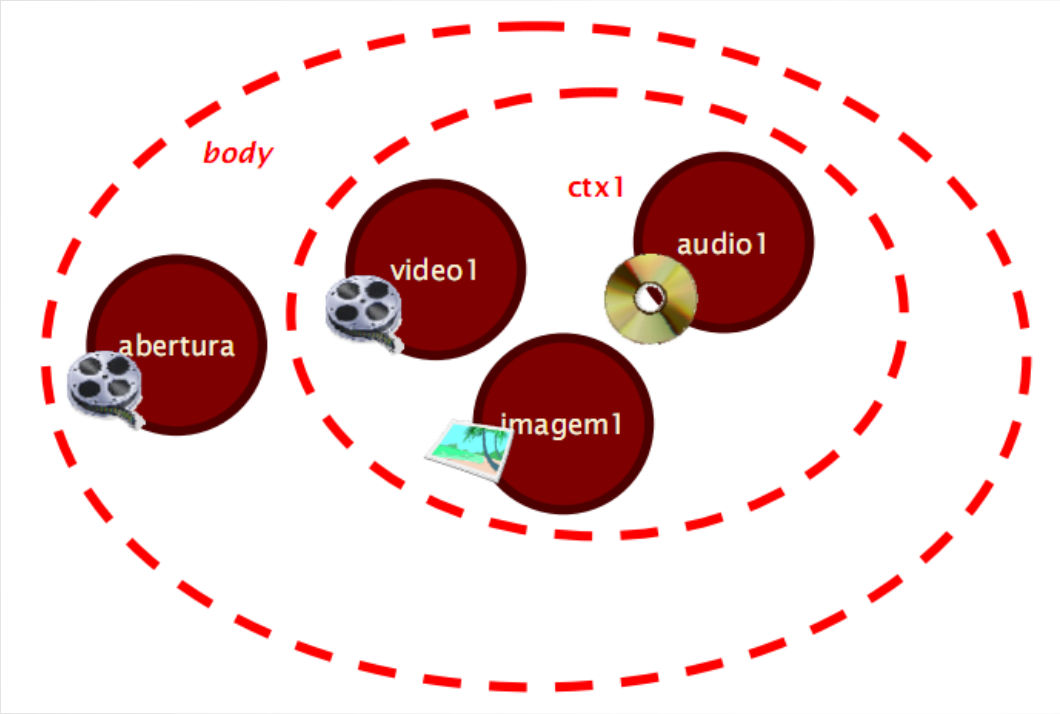
O quê? Nós



Nós de mídia

Esse conteúdo é representado através dos nós de mídia.

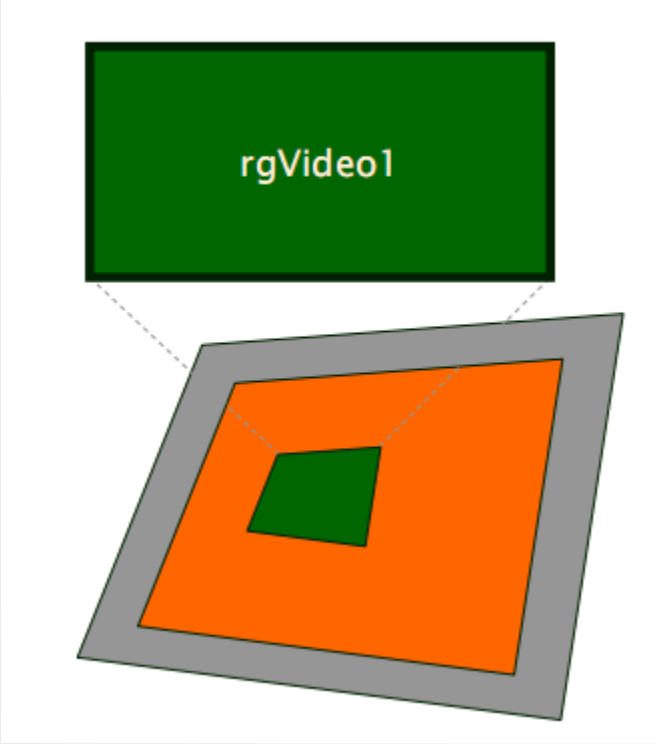
O quê? Nós



Todo nó de mídia é definido dentro de um contexto.

Representação de nós de mídia e de composição

Onde? Região



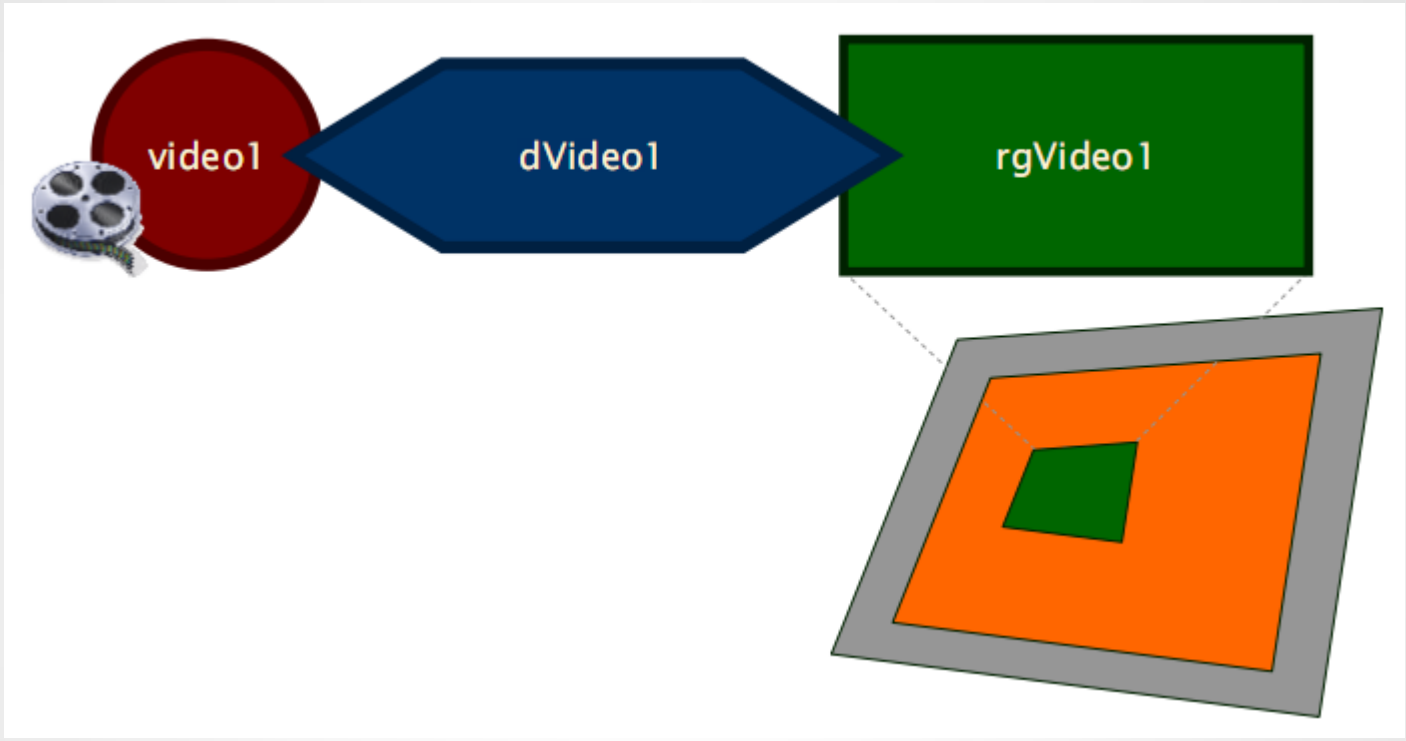
Representação de região utilizada pela mídia

Uma região indica a posição e as dimensões de uma área onde a mídia será apresentada

Descritor

- É nessa entidade que são dadas as características iniciais de apresentação.
- Define **onde** o objeto de mídia será apresentado.
  - Essa entidade é a base para o suporte a múltiplos dispositivos de exibição.
- Entidade que associa a mídia a uma região.

Descritor



Descritores fazem uma associação de uma mídia com várias regiões

Estrutura inicial do NCL

cabeçalho do arquivo NCL

```
1: <?xml version="1.0" encoding="ISO-8859-1"?>
2: <ncl id="exemplo01" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.ncl.org.br/NCL3.0/EDTVProfile
        http://www.ncl.org.br/NCL3.0/profiles/NCL30EDTV.xsd">
```

cabeçalho do programa

```
3: <head>
```

base de regiões

```
4:     <regionBase>
5:       <!-- regiões da tela onde as mídias são apresentadas -->
6:     </regionBase>
```

base de descritores

```
7:   <descriptorBase>
8:     <!-- descritores que definem como as mídias são
         apresentadas -->
9:   </descriptorBase>
```

1 - Cabeçalho básico do NCL ;

2 – As regiões da tela onde aparecerão os elementos visuais (regionBase);

3 – Como e onde os nós serão exibidos, através de descritores (descriptorBase);

Como? - Descritores

Parâmetros de exibição de mídia em uma determinada região

Introdução

- Especificam como os objetos de mídia serão exibidos;
- São definidos no cabeçalho do documento ncl;
- Todo nó de mídia que será apresentado

```
<head>
[...]
    <descriptorBase id="dbMain">
        <descriptor id="dUnicoDefault" region="rgFull" />
    </descriptorBase>
</head>
```

Importação de Bases de Descritores

- Importar Base de Descritores de outro documento NCL.

```
<descriptorBase>
    <importBase
        documentURI=“pasta/arquivo.ncl"
        alias=“apelido#id_do_elemento_importado”

    />
[...]
</descriptorBase>
```

Atributos básicos de Descritor

- *id: identificador único, utilizado nas referencias do descritor;
- region: identificador da região associada ao descritor;
- explicitDur: define a duração do objeto de mídia associado ao descritor. Ex.: “95.9s”(segundos);
- freeze: identifica o que acontece ao final da exibição do objeto de mídia. Ex.: “true”(congela o ultimo instante do video ao finalizar);

```
<descriptor
        id=“dExemplo”
        reggion=“rgTvExemplo”
        explicitDur=“25s”
        freeze=“true”
/>
```

Parâmetros do Descritor

- Elementos opcionais;
- As propriedades e seus respectivos valores dependem do tipo de mídia a ser apresentada
- Cada descritor pode conter diversos elementos. Ex.:

```
<descriptorBase>
        <descriptor id="dVideoPrincipal" region="rgFull">
             <descriptorParam name="nomeParam" value="valorParam" />
             <descriptorParam name="soundLevel" value="0.7" />
               [...]
        </descriptor>
</descriptorBase>
```

Parâmetros reservados para áudio

- “soundLevel”, “balanceLevel”, “trebleLevel”, “basslevel”: Valores entre 0 e 1 ou ‘0%’ e ‘100%’;
- No caso de “soundLevel”:
  - 0 ou 0% = MUTE;
  - 0.5 ou 50% = volume pela metade;

```
<descriptor id=“dAudioEnglish”>
     <descriptorParam name=“balancelevel” value=“0.5” />
     <descriptorParam name=“soundLevel” value=“0.7” />
</descriptor>
```

Parâmetros reservados para objetos visuais(vídeos, imagens)

- “location”: posição do objeto de mídia. (left,top) Ex.:

```
<descriptorParam name=“location” value=“150,300” />;
```

- “size”: dimensões do objeto de mídia. width e height separados por vírgula.
- “zIndex”: posição da região no eixo Z.
- “background”: cor de fundo quando a mídia não couber na região. (padrão: “transparent”)
- “fit”: valores possíveis: “fill”, “hidden”, “meet”, “meetBest” ou “slice”. (padrão: “meet”)

Parâmetros reservados para texto

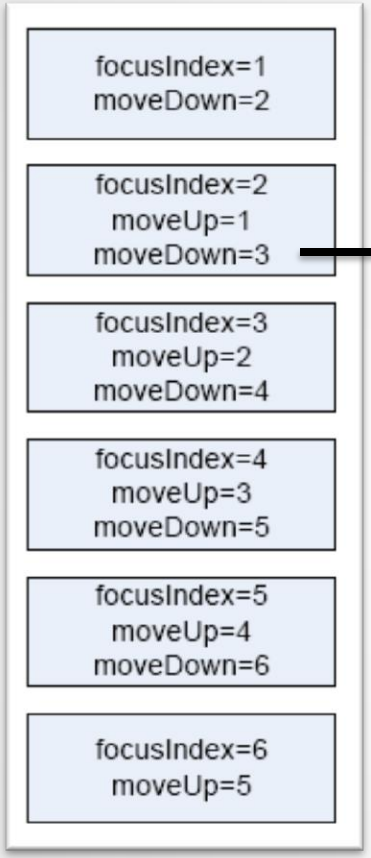
- “fontColor”: a cor da fonte (“white”, “blue”, “yellow”, etc). (padrão: “white”);
- “fontSize”: tamanho da fonte.
- “fontFamily”: lista com nome de fontes específicas ou genéricas.
- “fontVariant”: texto “normal” ou em “small-caps”. (padrão: “normal”).
- “fontWeight”: “normal” ou “bold”.

Navegação por teclas entre os objetos de mídia

- Deve-se, para tal evento, utilizar atributos no elemento `<descriptor />`
- Cada elemento deve conter um índice de foco.
- Utiliza-se este índice para indicar o destino do foco quando o usuário pressionar alguma seta
- Atributos utilizados no elemento "`<descriptor />`"

"*focusIndex*", "*moveLeft*", "*moveRight*", "*moveUp*", "*moveDown*", "*focusBorderColor*", "*focusBorderTransparency*", "*focusBorderWidth*", "*focusSrc*", "*focusSelSrc*", "*selBorderColor*"

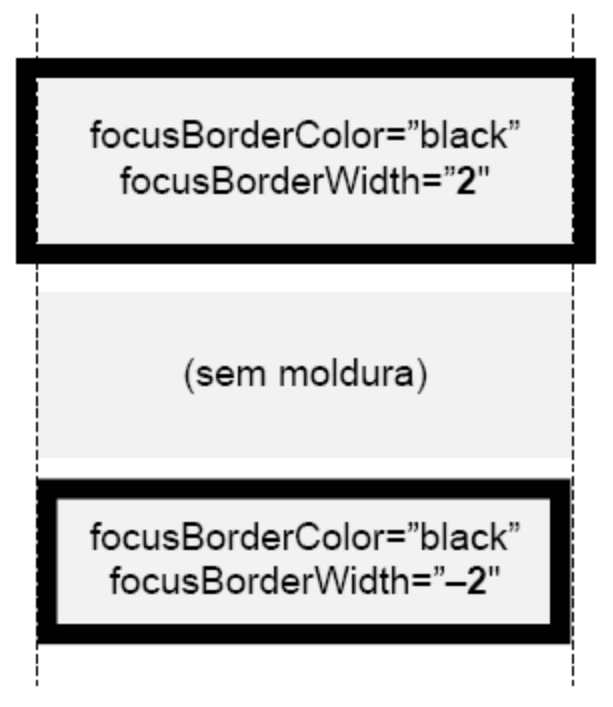
Exemplo de uso dos atributos de navegação



Atributos de descritor relacionados à navegação em um menu vertical de seis itens não-circular.

```
<descritor
        id=“dBtnOpc2”
        region=“rgbtn2”
        focusIndex=“2”
        moveDown=“3”
        moveUp=“1”
/>
```

Atributos para estilização de seleção atual



"`focusSrc`": permite a alteração do objeto de mídia.

Quando o usuário pressiona "OK", também é possível alterar a cor da moldura ou o próprio objeto de mídia. Através dos atributos: "`selBorderColor`" e "`focusSelSrc`"

Efeitos de transição

- São atributos no elemento “<descriptor>” que fazem referência a elementos “<transition>” de uma base de “<transitionBase>”;
- Os efeitos não são obrigatórios;
- Definidos dentro do cabeçalho do documento;

```
<transitionBase>
        <transition id=“tEfect1” type=“fade” subtype=“fadeFromColor”
                    fadeColor=“blue” />
</transitionBase>
<descriptorBase>
        <descriptor id=“dExemplo1” region=“rgExemplo” transIn=“tEfect1” />
</descriptorBase>
```

Objetos de mídia e Contexto

Elementos e atributos dos objetos de mídia e contexto

Introdução

- É representado pela tag “<media>”;
- Deve apresentar arquivo de mídia (“src”), descritor(“descriptor”) e identificador único(“id”);
- Atributo “type” opcional;

```
<body>
        ...
        <media id=“VideoPrincipal” src=“media/Principal.avi”
               descriptor=“dTvExemplo” />
</body>
```

Atributos de Objeto de Mídia

- “id”: identificador único, utilizado nas referencias ao objeto;
- “src”: caminho do objeto de mídia;

```
Src=“http://lims.ifpi.edu.br/images/myImage.png”
```

- “type”: define o tipo de mídia(opcional);

```
type=“image/png”
```

- “descriptor”: identificador do descritor que controle a exibição do objeto de mídia;

```
descriptor=“dLogotipoLIMS”
```

Atributo type – valores permitidos

Valor do type	Extensão de Arquivo do Atributo “src”
text	.htm, .html, .xhtml, .css, .xml, .txt
image	.bmp, .png, .gif, .jpeg, .jpg, .jpe
audio	.ua, .wav, .mp1, .mp2, .mp3, .mp4, .mpg4
video	.mp2, .mpeg, .mpg, .mpe, .mp4, .mpg4, .mng, .qt, .mov, .avi,
Application	.ncl, .lua, .xlt, .xlet, .class

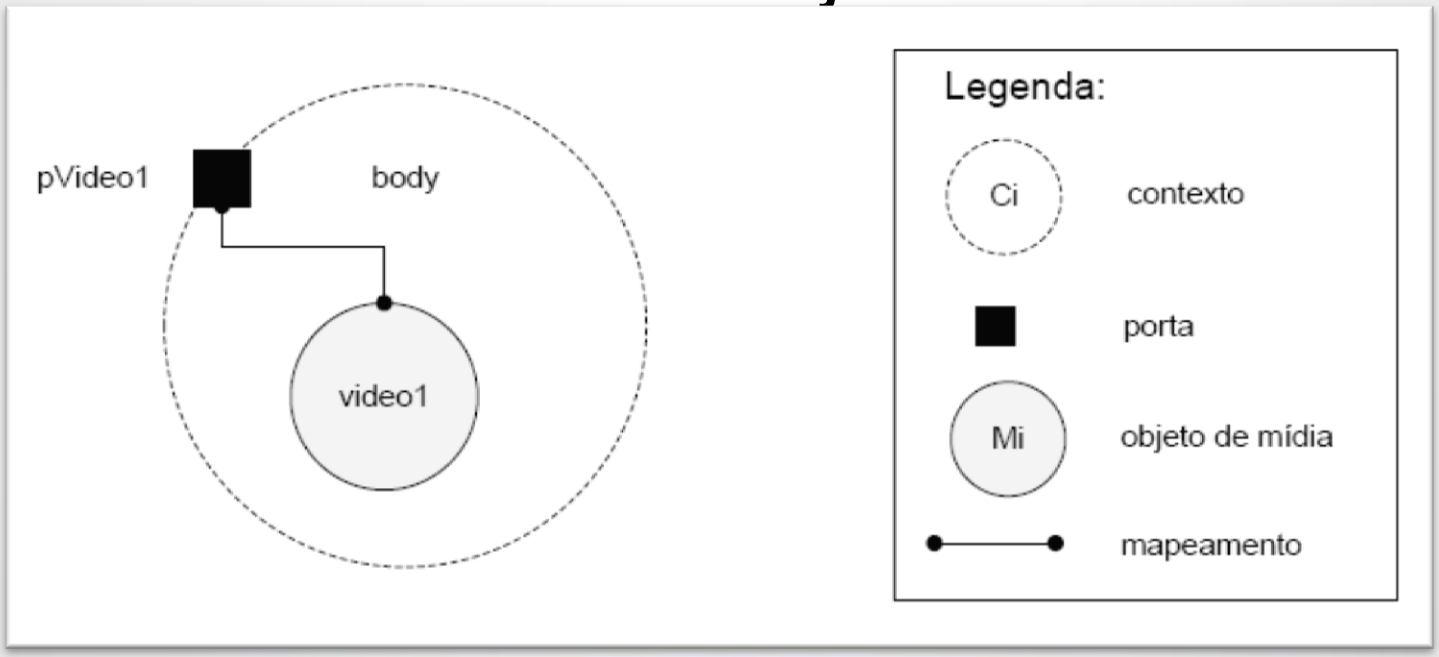
Contextos

- Agrupa objetos(de mídia, de contexto) e elos;
- Um exemplo é o elemento “<body>”;
- >”;

```
<body>
      ...
      <context id=“menu”>
            <!–portas-->
            <!–mídias, contextos -->
            <!- elos -->
      </context>
</body>
```

Portas

- Ponto de interface de um contexto que oferece acesso ao objeto do contexto.



Portas

- É necessário haver pelo menos uma porta no documento NCL, indicando qual o objeto(ou contexto) de mídia inicial.

```
<body>
        <port id="pInicio" component="videoPrincipal" />
        <port id="pInteratividade" component="imgInteratividade" />
        ...
</body>
```

Quando? – CONECTORES E ELOS

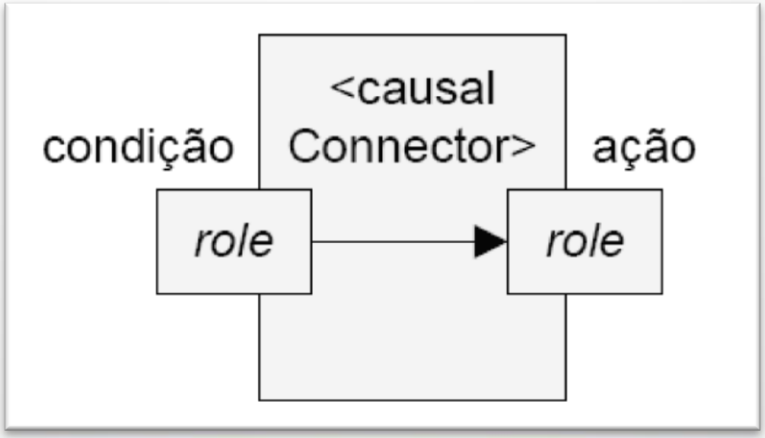
Definem o sincronismo e a interatividade entre os objetos de uma aplicação NCL.

Introdução

- Os conectores são definidos no cabeçalho do documento dentro de uma base de conectores;
- Os elos são inseridos no corpo, representados pela tag “<link>”;
- Um elo associa o objeto através de um conector.

Conectores

- Mecanismos de causalidade;
- Para uma ação ser disparada uma condição deve ser satisfeita;



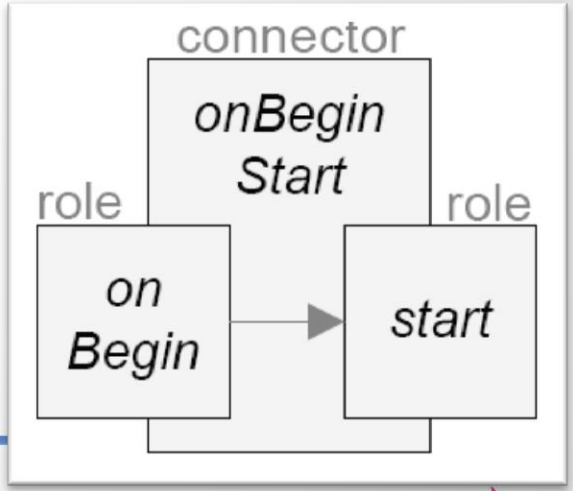
Elementos de Conectores

"<connectorParam>": define parâmetros, cujos valores são atribuídos pelos elos;

"<simpleCondition> e <compoundCondition>": define as condições de ativações do elo;

"<simpleAction> e <compoundAction>": define as ações que são realizadas.

```
<casualConnector id="onBeginStart">
        <simpleCondition role="onBegin" />
        <simpleAction role="start" />
</casualConnector>
```



Conectores

- “Quando condição/evento, então ação”.
- Cabe ao elo ligar os objetos ao papel;

Papéis de Condição

Papel	Descrição(quando o elo será ativado)
onBegin	Quando o objeto for iniciado
onEnd	Quando a apresentação for finalizada(natural ou intervenção)
onAbort	Quando a apresentação for abortada
onPause	Quando a apresentação for pausada
onResume	Quando a apresentação for retomada após uma pausa
onSelection	Quando uma tecla for pressionada enquanto um obj. estiver sendo apresentado

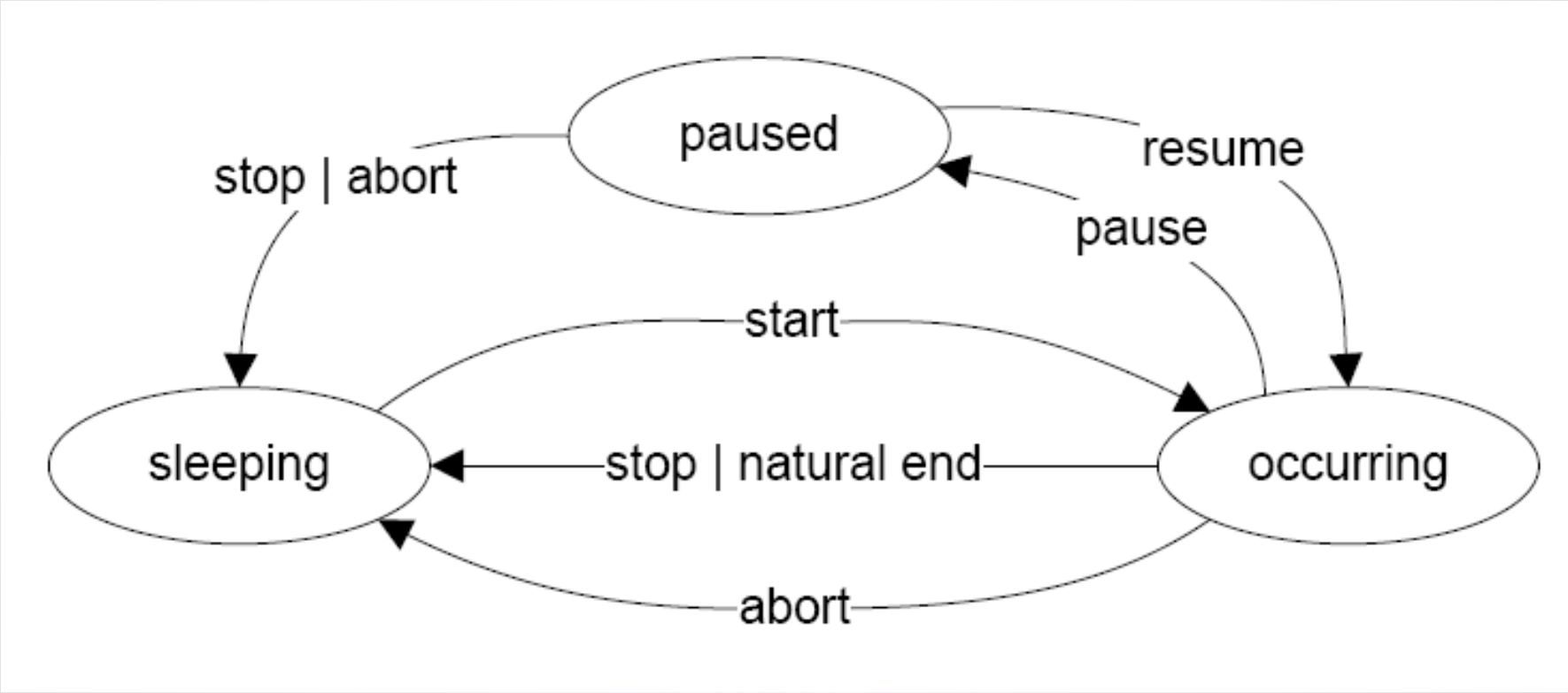
Papéis de Ação

Papel	Descrição (ação a ser realizada quando o elo for ativado)
start	Inicia a apresentação do objeto ligado a este papel
stop	Termina a apresentação do objeto ligado a este papel
Abort	Aborta a apresentação do objeto associado a este papel
Pause	Pausa a apresentação do objeto associado a este papel
Resume	Retoma a apresentação do objeto ligada a este papel (se estiver pausada)

<casualConnector>

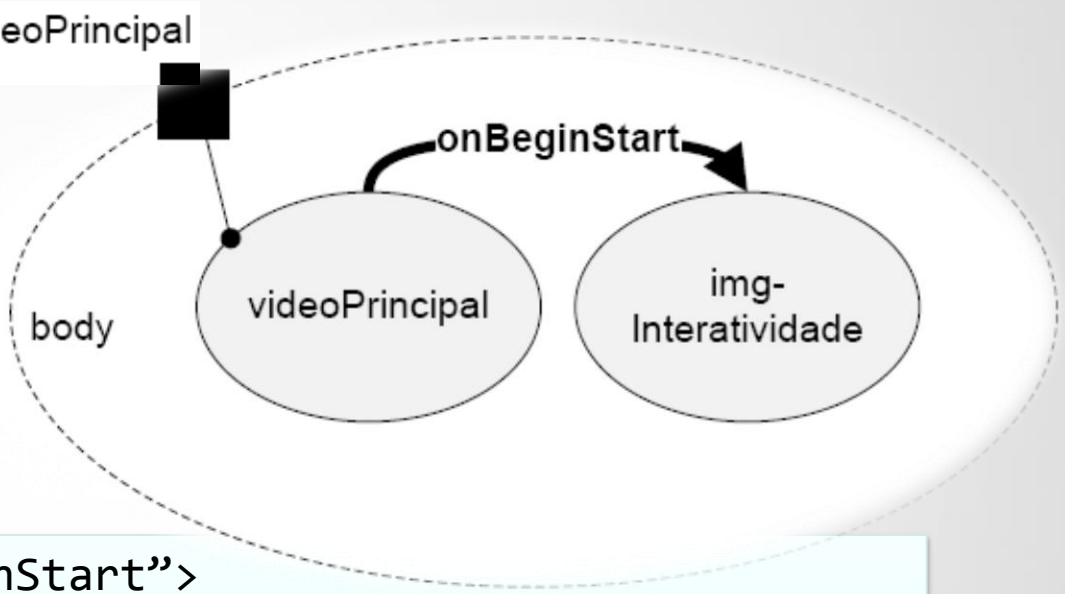
- Baseado em eventos;
- “eventos”: ocorrencia no tempo que pode ser instanciada ou duração mensurável;
- Tipos de Eventos:
  - Apresentação;
  - Selecção;
  - Atribuição;
  - Composição;

Máquina de estados de eventos



Elos

- Seu comportamento é definido pelo conector que u




```
<link xconnector="onBeginStart">
        <bind role="onBegin" component="videoPrincipal" />
        <bind role="start" component="imgInteratividade" />
</link>
```

Elementos de “<link>”

- “<bind>”: ligação entre objeto e papel;
- “<linkParam>”: define um parametro do elo;

```
<link xconnector=“id_do_conector”>
      <bind role=“id_do_papel_de_condição”
            component=“id_de_um_obj” />
      <bind role=“id_do_papel_de_acao”
            component=“id_de_um_objeto” />
</link>
```

Projeto em NCL

(Gerenciador de vídeo)

Aplicação em NCL

- Ferramentas
- Aplicação
- Requisitos
  - O que!
  - Onde!
  - Como!
  - Quando!

Ferramentas

- Composer IDE
- Eclipse
  - Plug-in ncl eclipse
- Ginga-NCL player
- Portal do software público

Aplicação

ginga NCL Lua

LIMS
Laboratory of Innovation on Multimedia Systems

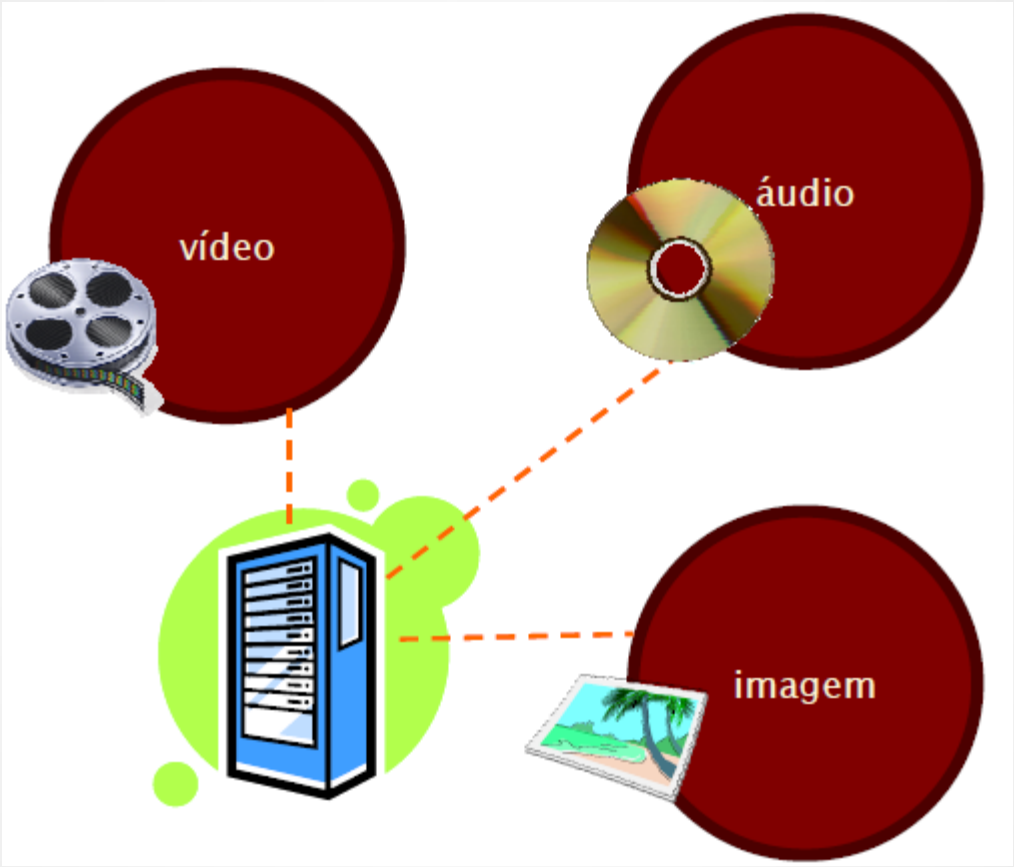
Requisitos

1. Definir as mídias (O que!)
2. Definir as regiões (Onde!)
3. Definir os descritores (Como!)
4. Definir o conectores e os elos (Quando!)

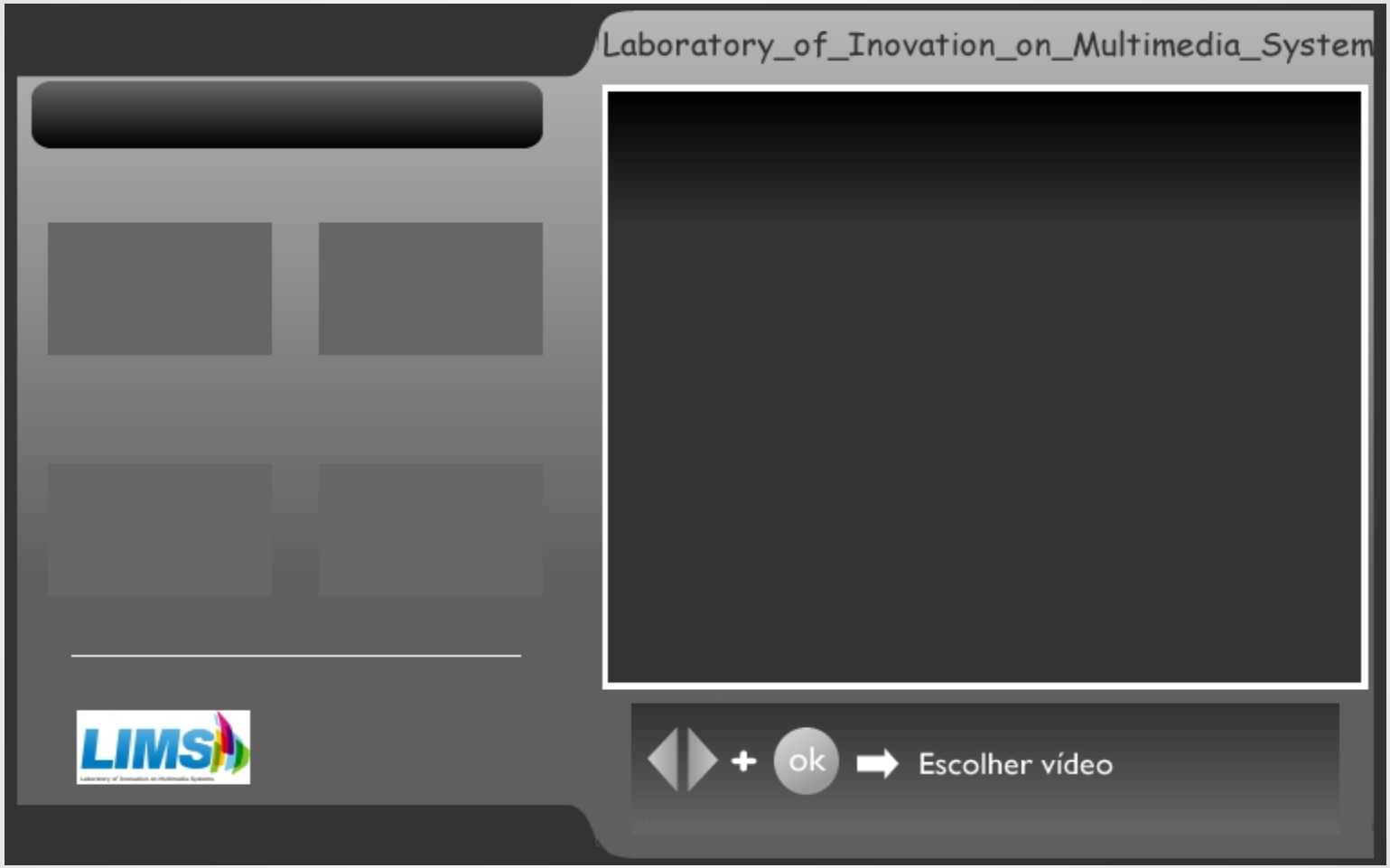
O que!

cabeçalho do arquivo NCL	1: <?xml version="1.0" encoding="ISO-8859-1"?> 2: <ncl id="exemplo01" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.ncl.org.br/NCL3.0/EDTVProfile http://www.ncl.org.br/NCL3.0/profiles/NCL30EDTV.xsd">	1
cabeçalho do programa	3: <head>	
base de regiões	4: <regionBase> 5: <!-- regiões da tela onde as mídias são apresentadas --> 6: </regionBase>	 2
base de descritores	7: <descriptorBase> 8: <!-- descritores que definem como as mídias são apresentadas --> 9: </descriptorBase>	 3
base de conectores	10: <connectorBase> 11: <!-- conectores que definem como os elos são ativados e o que eles disparam --> 12: </connectorBase>	8
	13: </head>	
corpo do programa	14: <body>	
ponto de entrada no programa	15: <port id="pInicio" component="ncPrincipal" interface="iInicio"/>	5
conteúdo do programa	16: <!-- contextos, nós de mídia e suas âncoras, elos e outros elementos -->	6 7 4
	17: </body>	
término	18: </ncl>	

O que!...



O que!....



ginga NCL Lua

LIMS
Laboratory of Innovation on Multimedia Systems

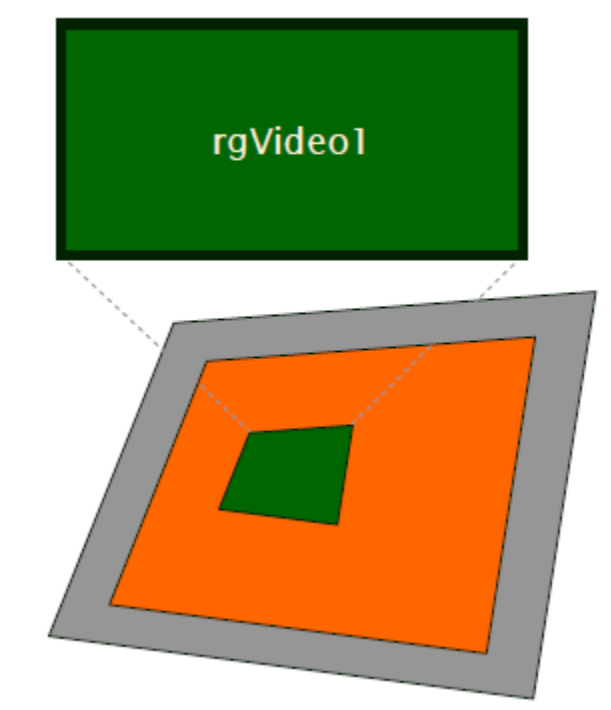
O que!....


Laboratory_of_Inovation_on_Multimedia_System
LIMS
ok
Escolher vídeo
Sair

Onde!

cabeçalho do arquivo NCL	1: <?xml version="1.0" encoding="ISO-8859-1"?> 2: <ncl id="exemplo01" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.ncl.org.br/NCL3.0/EDTVProfile http://www.ncl.org.br/NCL3.0/profiles/NCL30EDTV.xsd">	1
cabeçalho do programa	3: <head>	
base de regiões	4: <regionBase> 5: <!-- regiões da tela onde as mídias são apresentadas --> 6: </regionBase>	 2
base de descritores	7: <descriptorBase> 8: <!-- descritores que definem como as mídias são apresentadas --> 9: </descriptorBase>	 3
base de conectores	10: <connectorBase> 11: <!-- conectores que definem como os elos são ativados e o que eles disparam --> 12: </connectorBase>	8
	13: </head>	
corpo do programa	14: <body>	
ponto de entrada no programa	15: <port id="pInicio" component="ncPrincipal" interface="iInicio"/>	5
conteúdo do programa	16: <!-- contextos, nós de mídia e suas âncoras, elos e outros elementos -->	6 7 4
	17: </body>	
término	18: </ncl>	

Onde!...



Onde!...

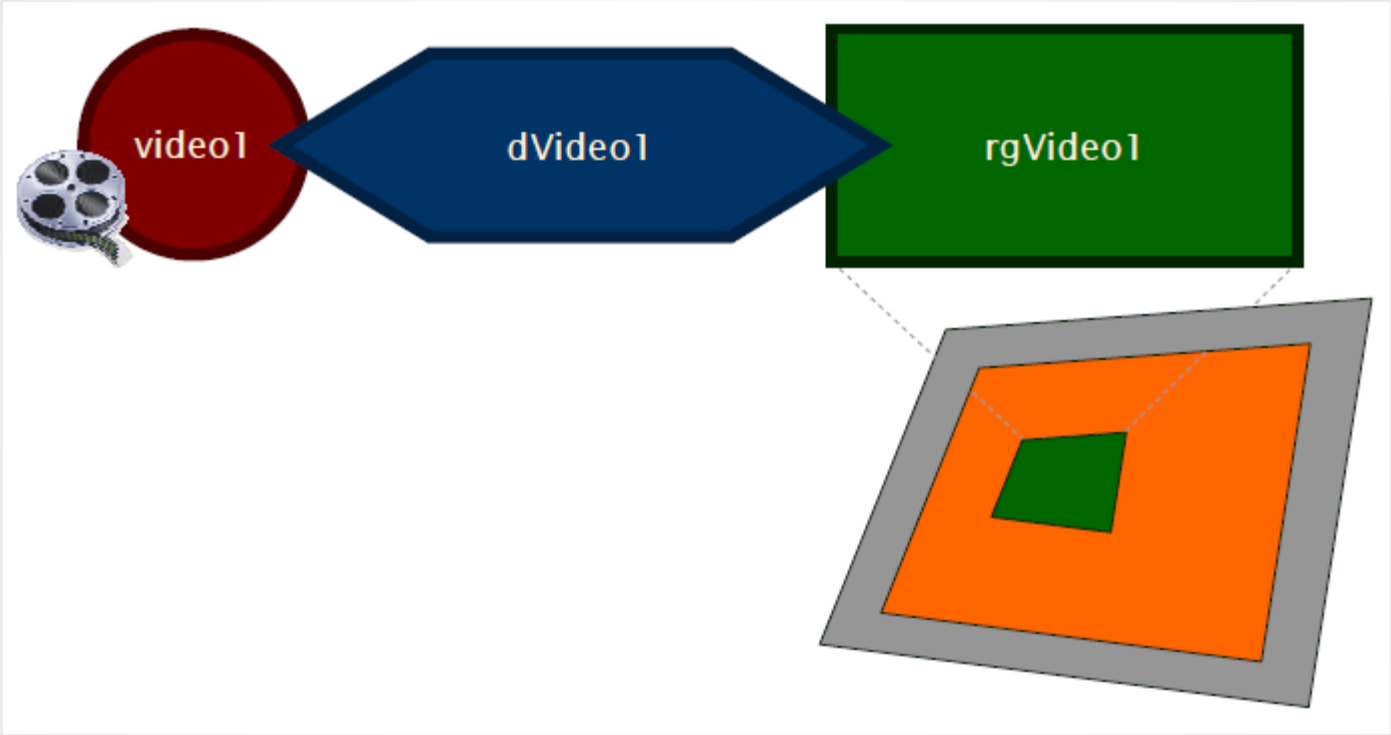
ginga NCL Lua

LIMS
Laboratory of Innovation on Multimedia Systems

Como!

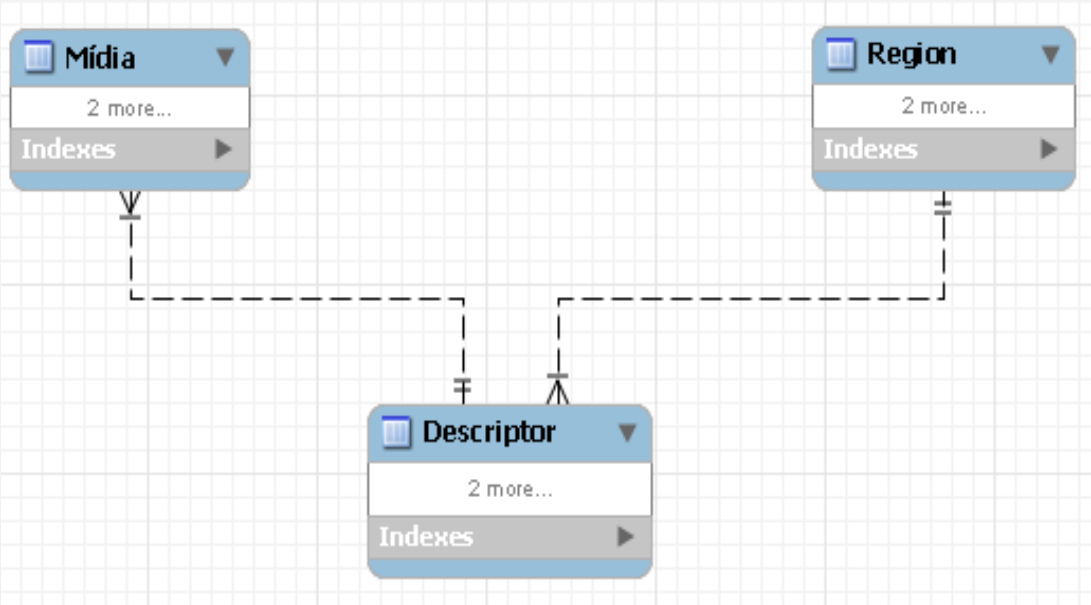
cabeçalho do arquivo NCL	1: <?xml version="1.0" encoding="ISO-8859-1"?> 2: <ncl id="exemplo01" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.ncl.org.br/NCL3.0/EDTVProfile http://www.ncl.org.br/NCL3.0/profiles/NCL30EDTV.xsd">	1
cabeçalho do programa	3: <**head**>	
base de regiões	4: <**regionBase**> 5: <!-- regiões da tela onde as mídias são apresentadas --> 6: </regionBase>	 2
base de descritores	7: <**descriptorBase**> 8: <!-- descritores que definem como as mídias são apresentadas --> 9: </descriptorBase>	 3
base de conectores	10: <**connectorBase**> 11: <!-- conectores que definem como os elos são ativados e o que eles disparam --> 12: </connectorBase>	8
	13: </head>	
corpo do programa	14: <**body**>	
ponto de entrada no programa	15: <**port** id="pInicio" component="ncPrincipal" interface="iInicio"/>	5
conteúdo do programa	16: <!-- contextos, nós de mídia e suas âncoras, elos e outros elementos -->	6 7 
	17: </body>	
término	18: </ncl>	

Como!...



Como!

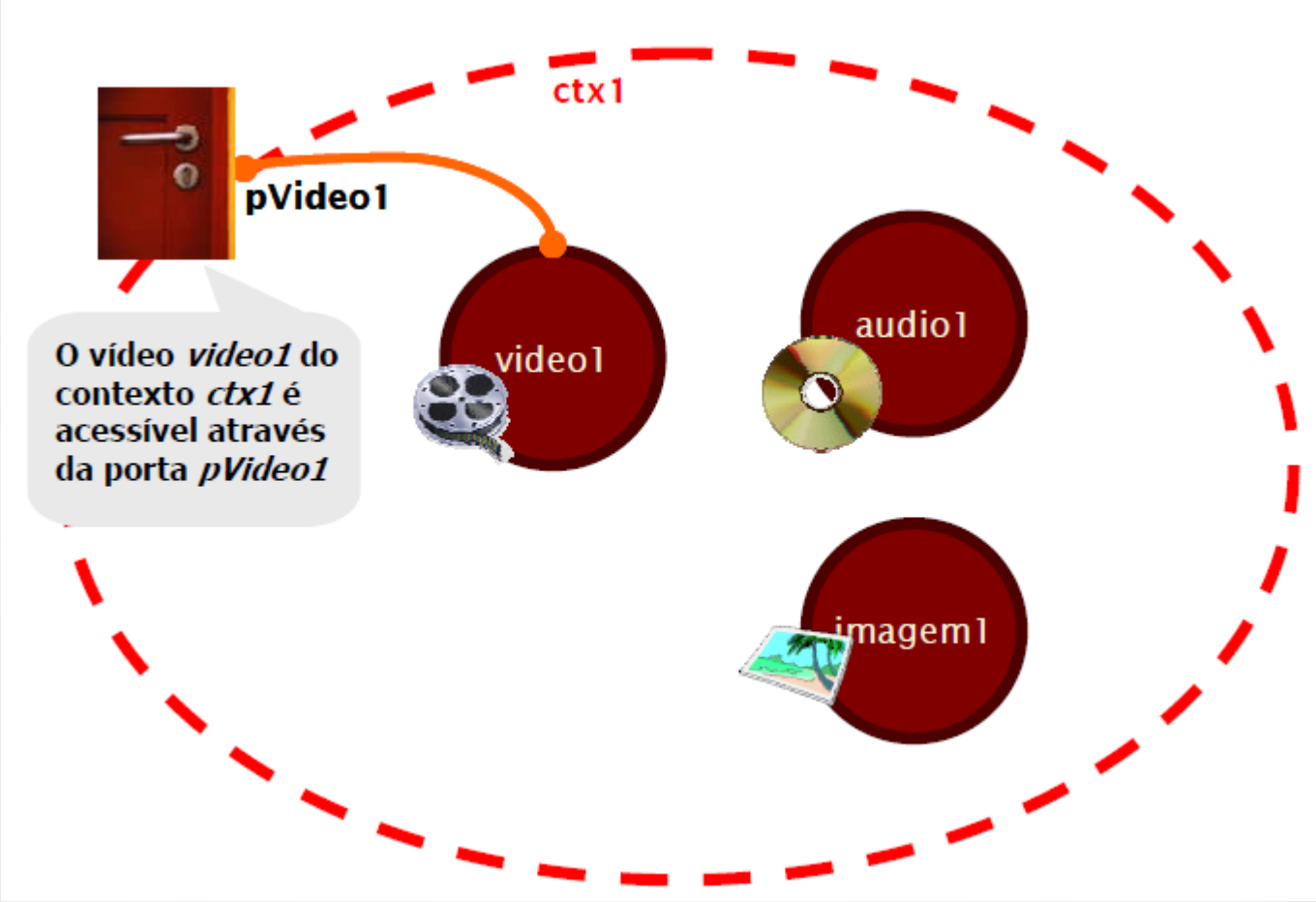
- Relacionamento



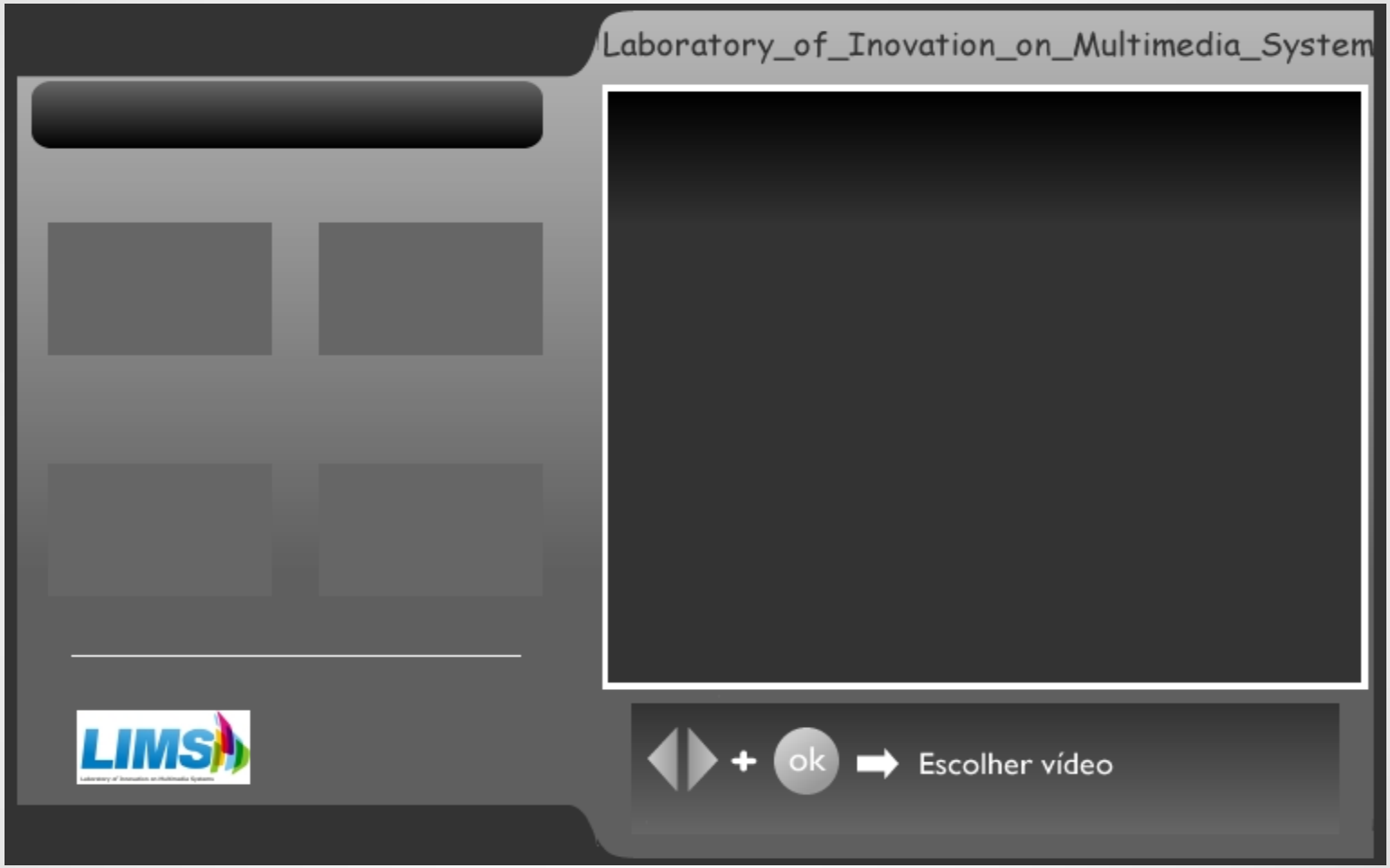
Quando!

cabeçalho do arquivo NCL	1: <?xml version="1.0" encoding="ISO-8859-1"?> 2: <ncl id="exemplo01" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.ncl.org.br/NCL3.0/EDTVProfile http://www.ncl.org.br/NCL3.0/profiles/NCL30EDTV.xsd">	1
cabeçalho do programa	3: <**head**>	
base de regiões	4: <**regionBase**> 5: <!-- regiões da tela onde as mídias são apresentadas --> 6: </regionBase>	2
base de descritores	7: <**descriptorBase**> 8: <!-- descritores que definem como as mídias são apresentadas --> 9: </descriptorBase>	d3 3
base de conectores	10: <**connectorBase**> 11: <!-- conectores que definem como os elos são ativados e o que eles disparam --> 12: </connectorBase>	8
	13: </head>	
corpo do programa	14: <**body**>	body
ponto de entrada no programa	15: <**port** id="pInicio" component="ncPrincipal" interface="iInicio"/>	5
conteúdo do programa	16: <!-- contextos, nós de mídia e suas âncoras, elos e outros elementos -->	6 7 4
	17: </body>	
término	18: </ncl>	

Quando!


ctx1
pVideo1
O vídeo *video1* do contexto *ctx1* é acessível através da porta *pVideo1*
video1
audio1
imagem1

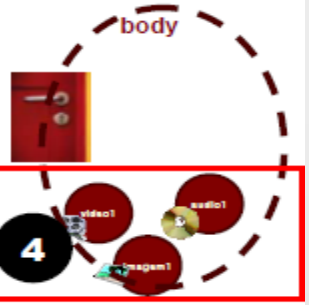
Quando!...



ginga NCL Lua

LIMS
Laboratory of Innovation on Multimedia Systems

Quando!

cabeçalho do arquivo NCL	1: <?xml version="1.0" encoding="ISO-8859-1"?> 2: <ncl id="exemplo01" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.ncl.org.br/NCL3.0/EDTVProfile http://www.ncl.org.br/NCL3.0/profiles/NCL30EDTV.xsd">	1
cabeçalho do programa	3: <**head**>	
base de regiões	4: <**regionBase**> 5: <!-- regiões da tela onde as mídias são apresentadas --> 6: </regionBase>	 2
base de descritores	7: <**descriptorBase**> 8: <!-- descritores que definem como as mídias são apresentadas --> 9: </descriptorBase>	 3
base de conectores	10: <**connectorBase**> 11: <!-- conectores que definem como os elos são ativados e o que eles disparam --> 12: </connectorBase>	8
	13: </head>	
corpo do programa	14: <**body**>	
ponto de entrada no programa	15: <**port** id="pInicio" component="ncPrincipal" interface="iInicio"/>	5
conteúdo do programa	16: <!-- contextos, nós de mídia e suas âncoras, elos e outros elementos -->	6 7 4
	17: </body>	
término	18: </ncl>	

Quando!

cabeçalho do arquivo NCL	1: <?xml version="1.0" encoding="ISO-8859-1"?> 2: <ncl id="exemplo01" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.ncl.org.br/NCL3.0/EDTVProfile http://www.ncl.org.br/NCL3.0/profiles/NCL30EDTV.xsd">	1
cabeçalho do programa	3: <head>	
base de regiões	4: <regionBase> 5: <!-- regiões da tela onde as mídias são apresentadas --> 6: </regionBase>	2
base de descritores	7: <descriptorBase> 8: <!-- descritores que definem como as mídias são apresentadas --> 9: </descriptorBase>	d3 3
base de conectores	10: <connectorBase> 11: <!-- conectores que definem como os elos são ativados e o que eles disparam --> 12: </connectorBase>	8
	13: </head>	
corpo do programa	14: <body>	
ponto de entrada no programa	15: <port id="pInicio" component="ncPrincipal" interface="iInicio"/>	5
conteúdo do programa	16: <!-- contextos, nós de mídia e suas âncoras, elos e outros elementos -->	6 7
	17: </body>	body 4
término	18: </ncl>	

Quando!...

```
<connectorBase>

    <causalConnector id="onBegin1StartN">
      <simpleCondition role="onBegin"/>
      <simpleAction role="start" max="unbounded" qualifier="par"/>
    </causalConnector>

    <causalConnector id="onSelectionStartN">
      <simpleCondition role="onSelection"/>
      <simpleAction role="start" max="unbounded" qualifier="par"/>
    </causalConnector>

    <causalConnector id="onKeySelectionStopN">
      <connectorParam name="keyCode" />
      <simpleCondition role="onSelection" key="$keyCode"/>
      <simpleAction role="stop" max="unbounded" qualifier="par"/>
    </causalConnector>

</connectorBase>
```

Quando!...

cabeçalho do arquivo NCL	1: <?xml version="1.0" encoding="ISO-8859-1"?> 2: <ncl id="exemplo01" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.ncl.org.br/NCL3.0/EDTVProfile http://www.ncl.org.br/NCL3.0/profiles/NCL30EDTV.xsd">	1
cabeçalho do programa	3: <head>	
base de regiões	4: <regionBase> 5: <!-- regiões da tela onde as mídias são apresentadas --> 6: </regionBase>	 2
base de descritores	7: <descriptorBase> 8: <!-- descritores que definem como as mídias são apresentadas --> 9: </descriptorBase>	 3
base de conectores	10: <connectorBase> 11: <!-- conectores que definem como os elos são ativados e o que eles disparam --> 12: </connectorBase>	8
	13: </head>	
corpo do programa	14: <body>	
ponto de entrada no programa	15: <port id="pInicio" component="ncPrincipal" interface="iInicio"/>	5
conteúdo do programa	16: <!-- contextos, nós de mídia e suas âncoras, elos e outros elementos -->	6 7 
	17: </body>	
término	18: </ncl>	

A linguagem Lua e suas aplicações

O Que é Lua

- Mais uma linguagem de script
  - = dinâmica
- Uma linguagem de Descrição de Dados
  - Não totalmente diferente de XML
- Única linguagem de programação desenvolvida fora do primeiro mundo a ter aceitação mundial
- Centenas de milhares de usuários
- Usada por Intel, Conectiva, Microsoft, LucasArts, Petrobras, etc.

Onde Lua é Desenvolvida

- Desenvolvida na PUC-Rio
  - desde 1993
- "Comitê" de três pessoas
  - Roberto Ierusalimschy, Luiz H. de Figueiredo, Waldemar Celes

Porque Lua

- Portabilidade
- Simplicidade
- Pequeno tamanho
- Acoplabilidade
- Eficiência

Portabilidade

- escrita em ANSI C, compila o mesmo código em todas as plataformas
- Unix, Windows (incluindo CE), Playstation II, OS/390, XBox, BeOS, DOS, Palm OS, EPOC, sistemas embutidos, etc.
- Núcleo é praticamente uma aplicação *free-standing*

Simplicidade

- Um único tipo de estrutura de dados
  - Tabelas
- Um único tipo numérico
  - tipicamente double
- Mecanismos ao invés de políticas
  - e.g., orientação a objetos

Pequeno Tamanho

- Menos de 200K
- bibliotecas independentes (e removíveis)

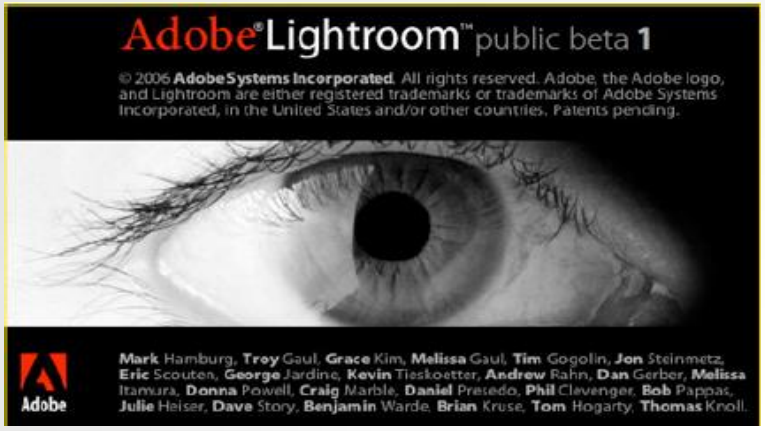
Acoplabilidade

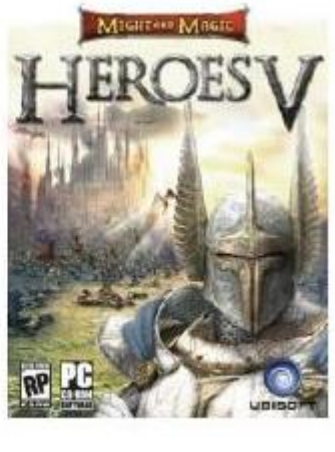
- Bi-direcional!
- Acoplada em C/C++, Java, Fortran, C#, Perl, Ruby, Ada, etc.

Eficiente

- Lua entre as mais rápidas no grupo de linguagens interpretadas com tipagem dinâmica

Onde é usada





TV Interativa se faz com ginga

Kepler Project

ginga NCL Lua

LIMS
Laboratory of Innovation on Multimedia Systems

Lua no desenvolvimento de jogos

- Motor de jogos
  - Simplifica o desenvolvimento
    1. Motor de Inteligência artificial (IA)
    2. Motor de física
    3. Motor gráfico
    4. Motor de som
    5. Linguagem de script
- Portabilidade
- Fornece funcionalidade
- Complexo e tem alto custo
- Reduz o custo do desenvolvimento

Mas por que utilizar a linguagem lua em jogos?

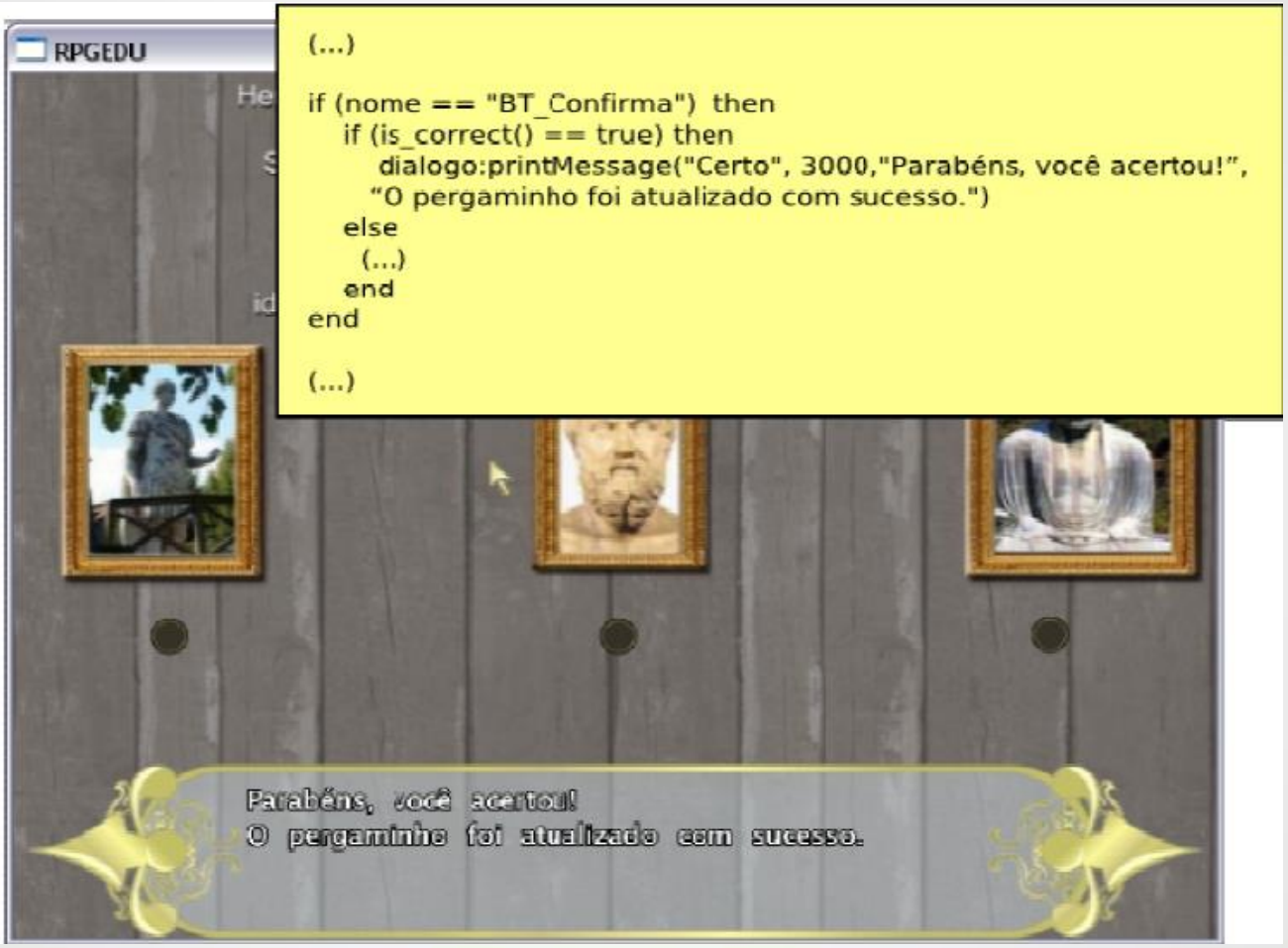
- Compilação
  - *código fonte é interpretado pelo programa em tempo de execução e não em tempo de compilação*
  - *Não precisar re-compilar o código principal*
  - *Maior flexibilidade no desenvolvimento da aplicação*
- Jogo do milhão, Soletrando
- Parametrização de seu conteúdo
- Apenas a parametrização do conteúdo não é a melhor solução

Mas por que utilizar a linguagem lua em jogos?

- Programação por não-programadores
- Desenvolvimento de *front-ends*
- Atualização dos *scripts*
- *Funcionamento interno do jogo*
- *Acesso aos códigos-fonte*

RPGDU

RPGDU...



ginga NCL Lua

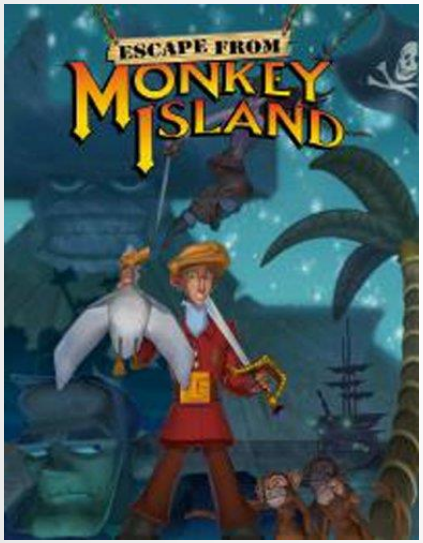
LIMS
Laboratory of Innovation on Multimedia Systems

Grim Fandango – *LucasArts*

- Adventure
  - Utiliza uma versão modificada de Lua 3.1 como linguagem de script

Escape from Monkey Island – *LucasArts*

- Adventure
  - Também utiliza uma versão modificada de Lua 3.1 como linguagem de script

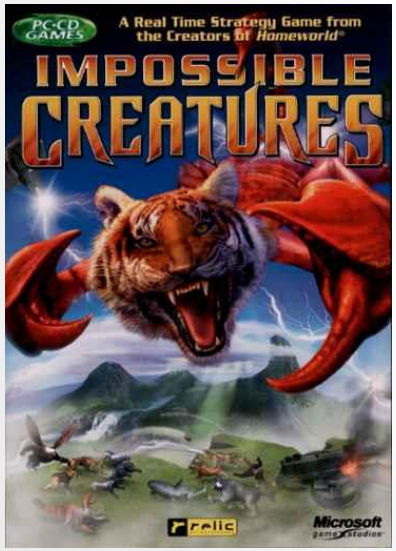


Psychonauts – *Double Fine*

- Action
  - Toda lógica do jogo implementada em Lua
  - Jogo controlado por entidades com scripts

Impossible Creatures – *Relic*

- Lua usada em
  - Controle de IA
  - Aparência de efeitos e de outros elementos gráficos
  - Determinação das regras do jogo
  - Edição dos atributos dos personagens
  - Debug em tempo real



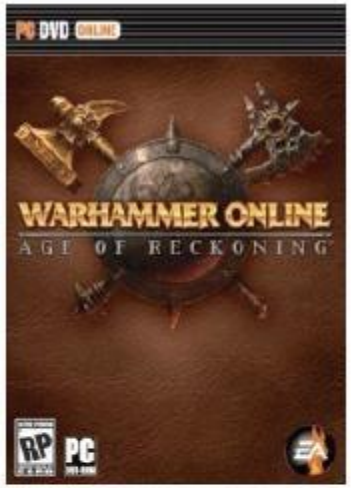
ginga NCL Lua

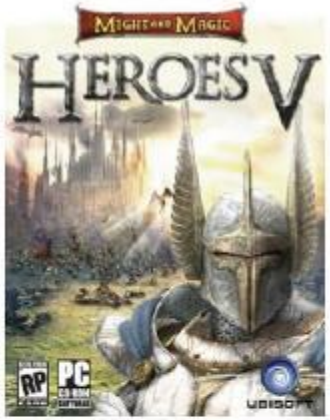
LIMS
Laboratory of Innovation on Multimedia Systems

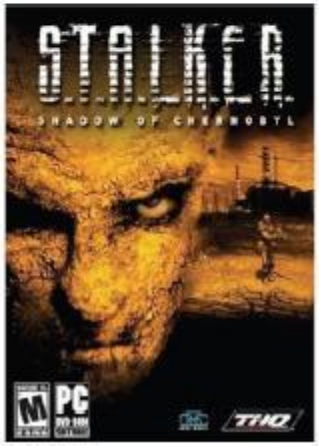
ginga NCL Lua

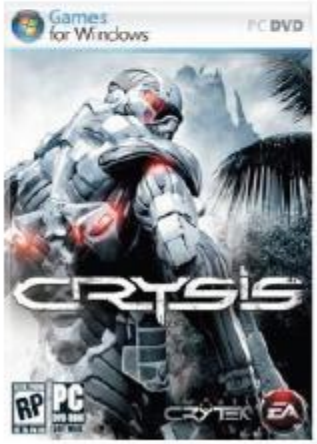
LIMS

Laboratory of Innovation on Multimedia Systems



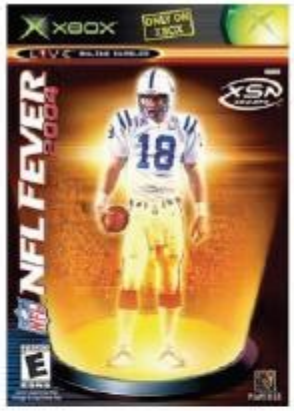


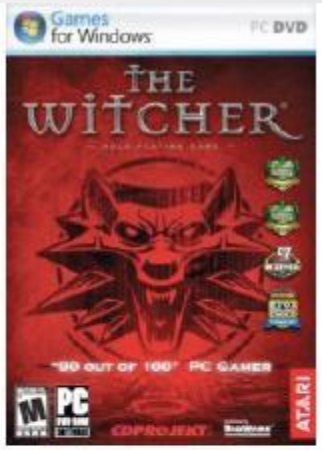


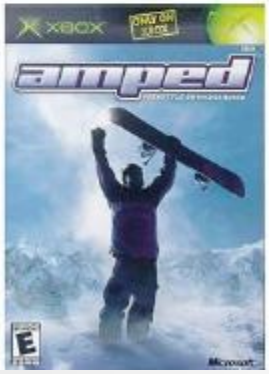


ginga NCL Lua

LIMS
Laboratory of Innovation on Multimedia Systems







ginga NCL Lua

LIMS
Laboratory of Innovation on Multimedia Systems

Lua no desenvolvimento para web

- Plataforma kepler - apresentação
  - extremamente portável e leve
  - é extensível
  - Kepler é um software livre
  - criado pela Fábrica Digital e pela PUC-RIO
  - Pré-requisitos
  - Kepler 1.1

Visão

- O Kepler não está tentando ser o novo PHP, derrubar o Python ou RoR
  - Leve e 100% nacional
- Alguma vez você já tentou colocar PHP ou Ruby em um dispositivo com 1Mb de RAM?
- O Kepler é Modular

Usos

- Fábrica digital
  - Publique!
  - Gerenciador de conteúdo para web
  - Mais utilizado do Brasil

- CISCO / PUC-RIO

Convergência digital

http://www.convergenciadigital.com.br/cgi/cgilua.exe/sys/start.htm?tpl=home

ginga NCL Lua

LIMS
Laboratory of Innovation on Multimedia Systems

Módulos

- CGILua
- LuaFileSystem
- LuaSocket
- Orbit (demonstrar...)
- Xavante
- LuaSql (demonstrar)
- LuaZip
- MD5
- ...

LuaRocks

- sistema de distribuição e gerenciamento para módulos *Lua*
- permite que você instale módulos Lua
  - Rocks
- contém informação sobre dependência de versão
- é uma aplicação de "puro Lua"

- Ginga
  - middleware padrão brasileiro para TV digital
- Wireshark
  - analisador de protocolos
- Snort
  - intrusion detection and prevention system
- nmap
  - rastreador de redes para segurança

Lightroom


Adobe Lightroom Beta – Library
Adobe Lightroom
Library | Develop | Slideshow | Print
Photo Library
Search
Browse by Shoot – 9
Browse Collections – 7
Browse Keywords – 0
View Options
Quick Develop
Import...
Export...

Introdução a linguagem Lua

Convenções Léxicas

- Palavras-chave

And break do else elseif end false
for function if in local nil not or
repeat return
then true until while

Convenções Léxicas...

- Diferencia minúsculas de maiúsculas

  and ~= AND ~= And

- Como convenção

  _VERSION

Itens léxicos

```
+   -   *   /   %   ^   #   ==   ~=   <=
>=   <   >   =   (   )   {   }   [   ]   ;
:   ,   .   ..   ...
```

Comentários

- -- Comentário de uma linha

- -- [[Esse é um comentário de múltiplas linhas]]

Variáveis

- São globais por padrão

```
mensagem
mensagem = “Hello, word“
print(mensagem)
```

- Variável local

```
local mensagem = “Hello, word“
print(mensagem)
```

- Campos de uma table

```
t[i]
```

- Palavras-chave

Atribuição

```
x = 3
x, y = 3, 2
x, y = y, x

x, y, z = 3, 2  -- z recebe nil
x, y = 3, 4 ,5, 6, 7   -- 5, 6 e 7
                          descartados
```

Valores e Tipos

- Dinamicamente tipada

```
local var = “oi“ -- contém uma
                    string
var = 3.14 -- Agora é um número
var = true -- e agora é um boolean
```

- *Valores de primeira classe*

Tipos

- Nil
- Boolean
- Number
- String
- Function
- Table
- Userdata
- Thread

Nil e Boolean

- Representa a ausência de um valor útil
- Representa true e false
- Variáveis não inicializadas têm valor nil
- Qualquer valor diferente de nil e false è verdadeiro
  - *String* vazia ("") e zero (0) são considerados verdadeiros

Number

- Números
- Equivale a um double (por padrão)
- Sem problema para representar inteiros

String

- *Cadeias de caracteres literais*
- Podem ser delimitadas através do uso de aspas simples ou aspas duplas
  - local mensagem = ‘oi’
  - local mensagem = “oi”
- Podem conter as seguintes seqüências de escape no estilo de C
  - local mensagem = “oi\n123”
- Formato longo
  - local mensagem = [[Cadeias literais longas]]
- Imutáveis

Coerção e concatenação

- Conversão automática

```
local str1 = "As armas "
local str2 = "e os barões assinalados"
  local concatenada = str1..str2
print ("1.234" + 4.321) --> 5.555
print ("Valor: "..171) --> "Valor:171"
```

Function

- Representa as funções
- Valores de primeira classe
- Pode ter nenhum ou vários de parâmetros
- Pode retornar nenhum ou vários valores

```
foo = 'Cadeia de Carac...'
function foo_1(n)
   return 1
end

foo_2 = function(n)
   return 1, 2, 3
end
```

Userdata

- Permite que dados C arbitrários possam ser armazenados em variáveis Lua.

Thread

- Representa fluxos de execução independentes
- é usado para implementar co-rotinas

Table

- Implementa arrays associativos
- Representar arrays comuns, tabelas de símbolos, conjuntos, registros, grafos, árvores, etc.
- Existem várias maneiras convenientes de se criar tabelas em Lua
- Podem conter funções

```
local t = { }
local t={4, 'lua', false}
→ t[1]=4, t[2]='lua', t[3]=false

local t = { nome = "Fulano", idade = 33,
            pais = { pai = "Ivo", mae = "Ana"}}
   print (t["nome"]) --> "Fulano"
   print (t.nome) --> "Fulano"
   print (t["pais"]["pai"]) --> "Ivo"
   print (t.pais.mae) --> "Ana"
```

Expressões

- aritméticas
  - op: + - * / ^ %
  - ex: 1+1 (v / 2)^2 -x
  - número -> número

- de concatenação
  - op: ..
  - ex: “João” .. “ e “ .. “Maria”
  - *string -> string*

Expressões

- lógicas
  - op: and or not
  - ex: 1 and 2 → 2
    - 1 and false → false
    - not val → false
    - 1 or 2 → 1
- *relacionais*
  - op: < > <= >= == ~=
  - ex: 1<=2 “a”~=”a” 1==”1” x>y
  - valor -> *boolean*

Estruturas de Controle

```
local x = 1
```

- ```
  while exp do
      bloco
  end
  ```

  ```
  while x < 10 do
      print(x)
      i = i + 1
  end
  ```

- ```
  repeat
      bloco
  until exp
  ```

  ```
  repeat
      print(x)
  until x < 10
  ```

- ```
  if exp then
      bloco
  elseif exp then
      bloco
  else
      bloco
  end
  ```

  ```
  if x == 1 then
      print('x = 1')
  elseif x==2 then
      print(x==2)
  else
      print('x ~= 1e x~= 2')
  end
  ```
  ```

O comando for e suas variações

- Numérica

```
for i = 1, 10, 1 do
  if i % 2 == 0 then
      print (tostring(i).."é par.")
  else
      print (tostring(i).." é ímpar.")
  end
end
```

O comando for e suas variações...

- Genérica

```
hero = { name = "Freddy Hardest",
         speed = 3,
         strength = 12,
         intelligence = 7 }

for k, v in pairs (hero) do
    print (k.."-->".. v)
end
```

Biblioteca Básica

- funções de uso geral

- assert, dofile, error, **_G**, ipairs, next, pairs, pcall, **print**, **tonumber**, **tostring**, **type**, unpack, **_VERSION**

Biblioteca math

- funções trigonométricas
- exponenciação e logaritmo
- arredondamento
- max, min
- randomização

- **math.sin**, **math.pi**, math.log, **math.pow**, math.sqrt, math.random

Biblioteca table

- tabelas como *arrays*
- inserção e remoção
- ordenação
- concatenação

- **table.insert**, **table.remove**, table.sort, table.concat, table.maxn

Biblioteca string

- funções básicas
- *pattern-matching*

- **string.sub**, string.format, string.rep, string.find, **string.lower**, **string.len**

Outras bibliotecas

- Debug -> Depuração
- File -> Entrada e saída
- Os -> Funções de sistema e data e hora

Beginning Lua Programming
Wrox, 2007

Lua 5.1 Reference Manual
Lua.org, 2006

Game Development with Lua
Charles River Media, 2005

入門Luaプログラミング (単行本)
Softbank Creative, 2007

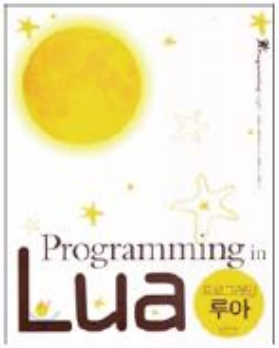
ginga NCL Lua

LIMS
Laboratory of Innovation on Multimedia Systems

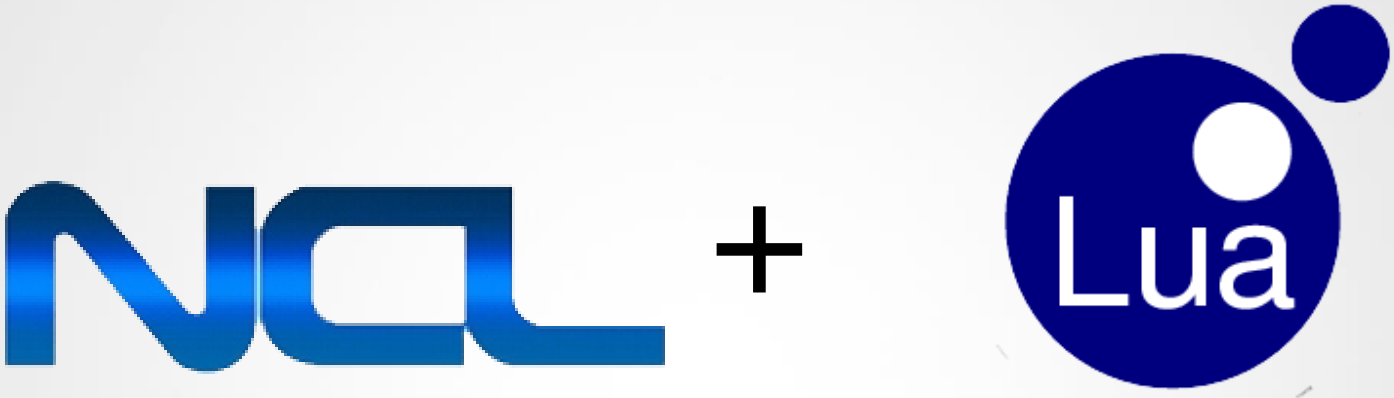
Programmieren mit Lua
Open Source Press, 2006

Programming in Lua, 2nd edition
Lua.org, 2006

프로그래밍 루아
Insight, 2007



Lua程序设计
PHEI, 2008


NCL
+
Lua

+

Declarativo

- Mais alto nível de abstração;
- Resultam em uma declaração de um resultado desejado;
- “o que fazer”;

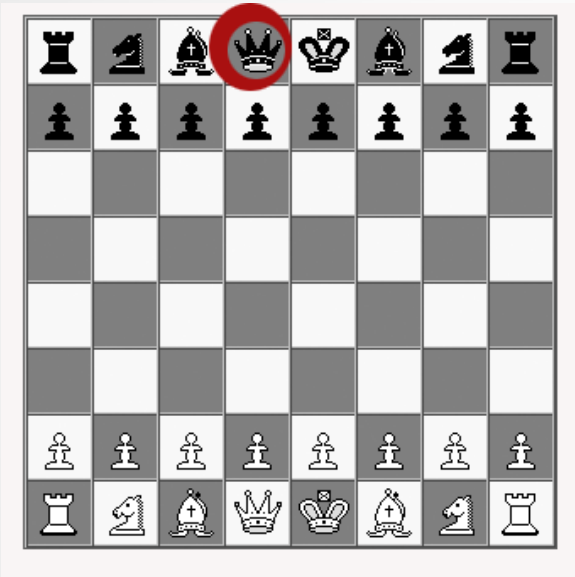
Ex: NCL, HTML e SQL

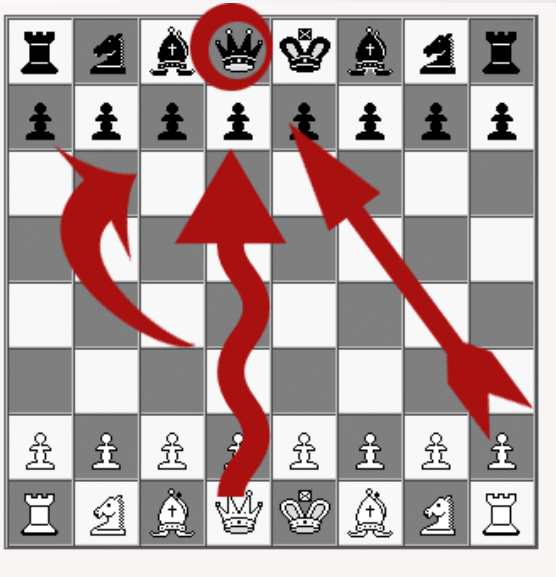
Não Declarativo (Imperativo)

- Podem seguir diferentes modelos;
- Implementação algorítmica;
- “como fazer”;
- Relação direta com a CPU;
- Programador Especialista;

Ex: LUA, Java, C++

NCL + Lua





“O que fazer”

“Como fazer”

Ambientes de aplicações para receptores fixos e móveis

Middleware	Sistema de TVD	Ambiente Declarativo	Ambiente Procedural
ACAP	Americano/ATSC	ACAP-X [ATSC A-101 2005] (linguagem declarativa = XHTML like; linguagem não-declarativa = ECMAScript)	ACAP-J [ATSC A-101 2005] (linguagem não-declarativa = Java
MHP	Europeu/DVB-T	DVB-HTML [ETSI TS 102 812 V1.2.2 2006] (linguagem declarativa = XHTML like; linguagem não-declarativa = ECMAScript)	MHP [ETSI TS 102 812 V1.2.2 2006] (linguagem não-declarativa = Java
ARIB-BML	Japonês/ISDB-T	ARIB – BML [ARIB B-24 2004] (linguagem declarativa = BML (XHTML like; linguagem não-declarativa = ECMAScript)	Opcional (GEM [ETSI TS 102 819 V1.3.1 2005] like); não implementado)
Ginga	Brasileiro/SBTV D	Ginga-NCL [ABNT NBR 15606-2 2007] (linguagem declarativa = NCL; linguagem não-declarativa = Lua)	Ginga-J (linguagem não-declarativa = Java

Todo middleware de ambiente declarativo ou procedural deve dar suporte as seguintes requisitos:

- Suporte a sincronização;
- Suporte a múltiplos dispositivos;
- Suporte a edição ao vivo;

Relação entre Objetos de Mídia

- Um objeto com código imperativo deve ser escrito em um arquivo separado do documento NCL, que apenas o referencia.
- Relacionamentos são independentes do tipo de mídia.
  - Através de Elos: <link>;

Exemplo:

```
<media id="myvideo" src="video.mpg"/>
<media id="mynclua" src="nclua.lua"/>

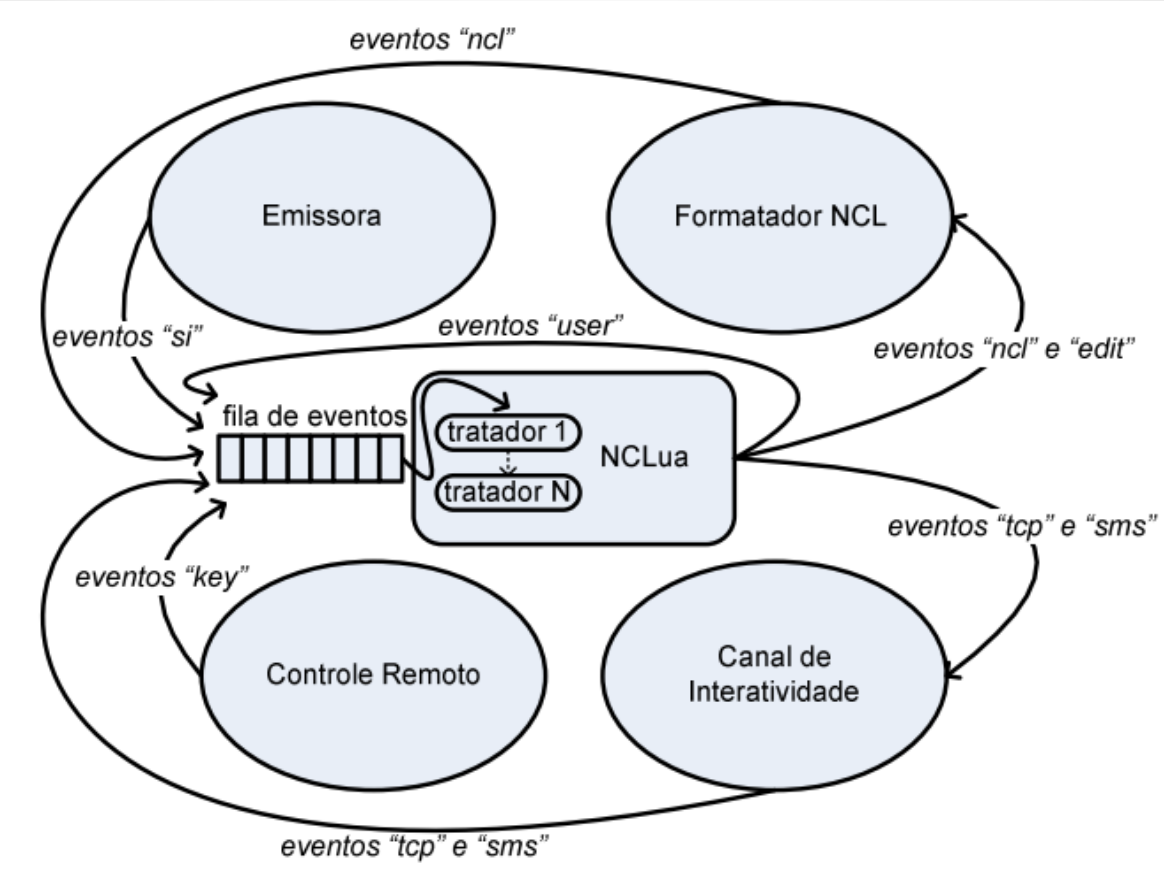
<link>
   <bind role="onBegin" component="myvideo"/>
   <bind role="start" component="mynclua"/>
</link>
```

- **Módulo event**
  - Permite que objetos NCLua se comuniquem com o documento NCL;
- **Módulo canvas**
  - Oferece a funcionalidade para desenhar objetos gráficos;
- **Módulo settings**
  - Oferece acesso às variaveis definidas no objeto settings;
- **Módulo persistent**
  - Exporta uma tabela persistente entre execuções de objetos imperativos;
- **Módulo ncledit**
  - Capaz de editar documentos NCL

Paradigma Orientado a Eventos

- O modelo de execução de um NCLua é orientada a eventos;
- O módulo event é a mais importante extensão;
- O script NCLua não é nada mais que um *tratador de eventos*;
  - Apenas um evento é tratado por vez;
  - Processamento deve ser rápido;

Paradigma Orientado a Eventos



Paradigma Orientado a Eventos

```
-- initialization
...
function hdlr (evt)
   -- NCL events
   if evt.action == 'start' then
      ...
   end
   -- key events
   if key.value == '1' then
      ...
   end
end
event.register(hdrl)
```

Paradigma Orientado a Eventos

```
evt = {
class =‘key’
type = ‘PRESS’
key = ‘RED’
}
```

A função tratadora pode receber um evento indicando que a tecla vermelha do controle remoto foi pressionada pelo telespectador

Representação de evento em NCLua.

Paradigma Orientado a Eventos

```
event.post = {
class =‘ncl’
type = ‘presentation’
action = ‘stop’
}
```

A função tratadora pode receber um evento indicando que a tecla vermelha do controle remoto foi pressionada pelo telespectador

Representação de evento em NCLua.

Classes de Eventos

Comunicação NCL

- Class: 'ncl'
- Type: 'presentation', 'attribution'
- Action: 'start', 'stop', 'set', ...
- Transition: 'pauses', 'set', ...
- Area: '', 'fim', 'fase1', ...

Classes de Eventos

- Teclas do controle
  - Representa o pressionamento de teclas do controle remoto pelo usuário;

    class: 'key'
    type: 'press' e 'release'
    key: 'RED', 'A', '1', ...

  { class='key', type='press', key='RED' }

- Eventos internos
  - Através dessa classe, aplicações podem estender sua funcionalidade criando seus próprios eventos;

  class: 'user'
  any: any
  { class='user', data={k1=v1,k2=v2} }

Classes de Eventos

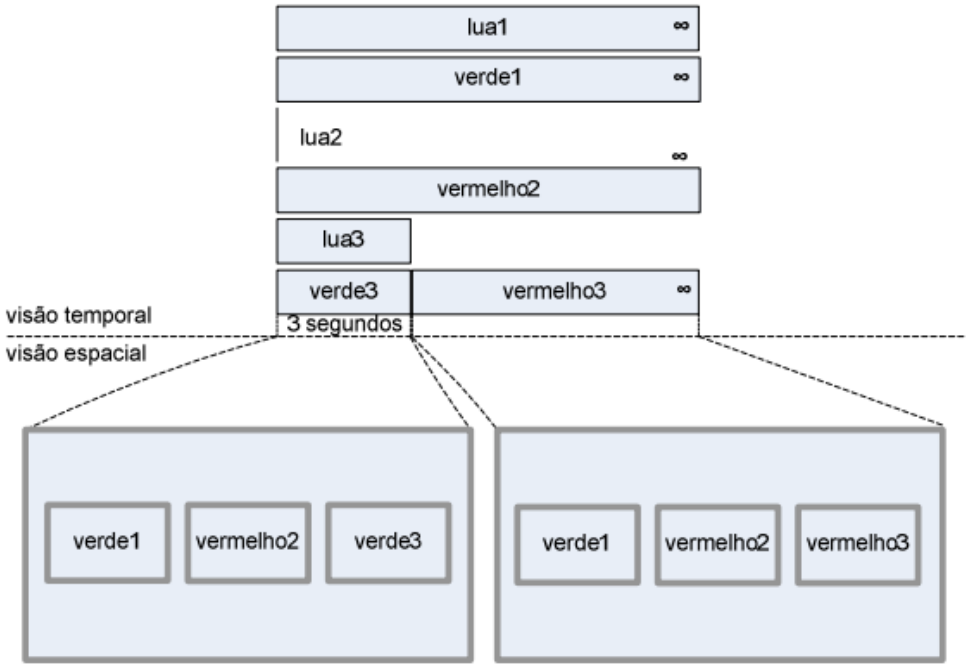
• Classe tcp: Permite acesso ao canal de interatividade por meio do protocolo tcp.

• Classe sms: Usada para envio e recebimento de mensagens SMS em dispositivos móveis.

• Classe edit: Permite que os comandos de edição ao vivo sejam disparados a partir de scripts NCLua.

• Classe si: Provê acesso a um conjunto de informações multiplexadas em um fluxo de transporte e transmitidas periodicamente por difusão.

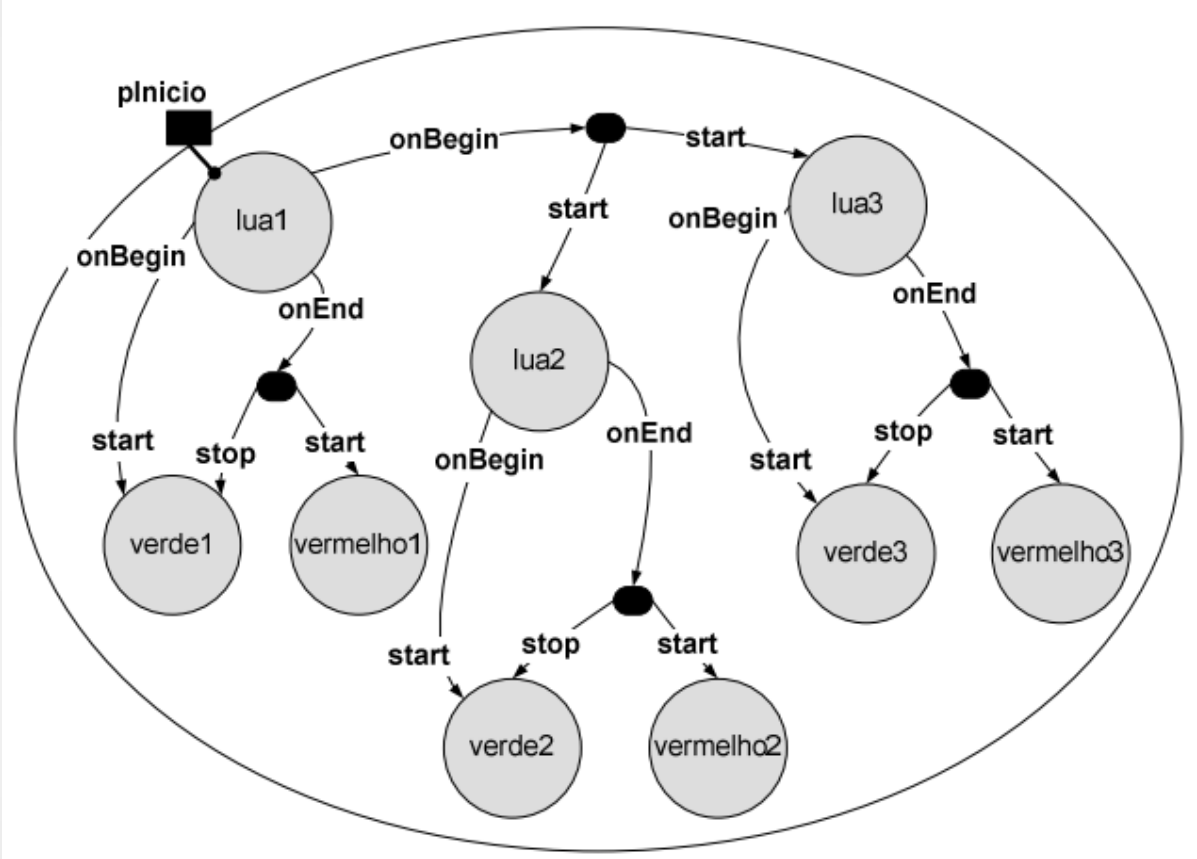
Exemplo 1



- Três nós NCLua são disparados;
- O primeiro não trata eventos;
- O segundo notifica seu fim natural ao receber um evento (start);
- O terceiro cria um timer de 3 segundos para notificar seu fim natural;
- Botões identificam seus estados;

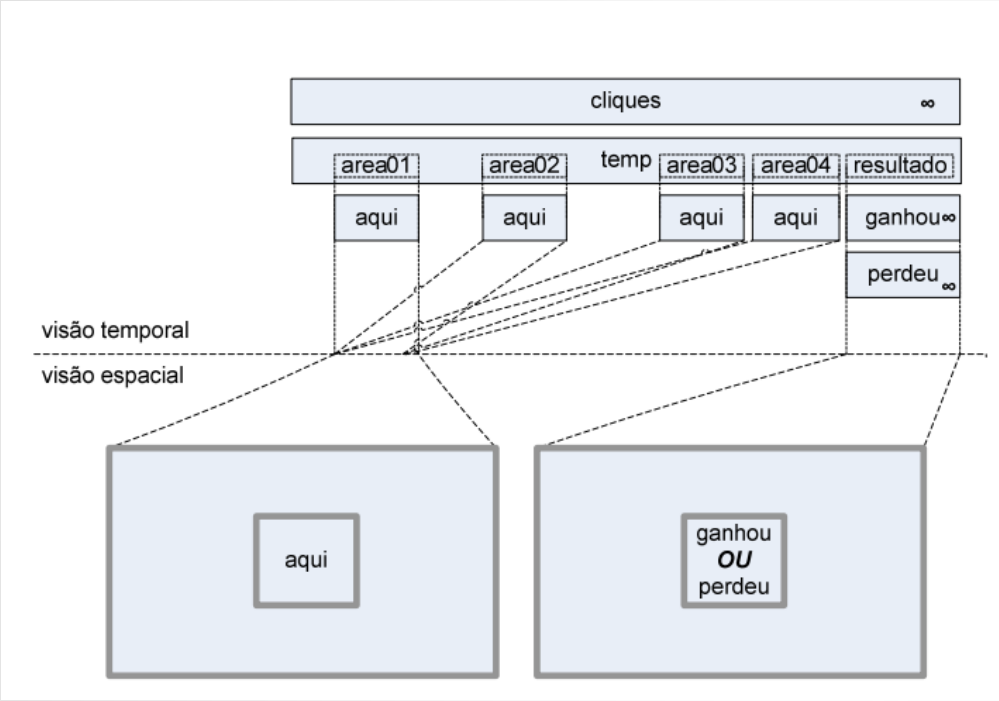
Visão temporal e espacial do Exemplo 1

Exemplo 1



Visão estrutural do Exemplo 1

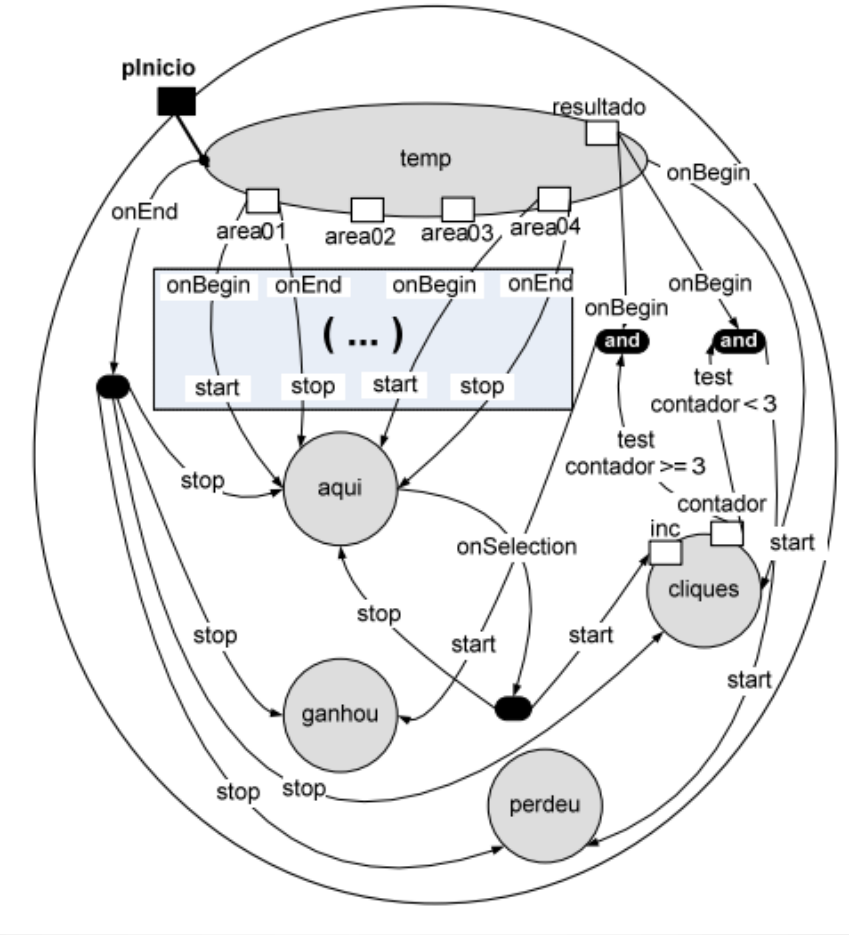
Exemplo 2



- Botão “Click it” aparece diversas vezes durante o vídeo;
- Conta o número de vezes que o usuário seleciona o botão;
- Em NCL puro: explosão de elos;
- Em Lua: uma variável e um método;
- Uso de uma propriedade para comunicação;

Visão temporal e espacial do Exemplo 2

Exemplo 2



Visão estrutural do Exemplo 1

Módulo Canvas

Operações gráficas durante a apresentação de uma aplicação

Construtores

- *canvas:new (image_path : string)*
  - Construtor utilizado para gerar um novo canvas cujo conteúdo é a imagem passada através do argumento image_path.
- *canvas:new (width, height : number)*
  - Construtor utilizado para gerar um novo canvas com todos os pixels de cor transparente.

Funções

- *canvas:attrSize() -> width, height : number*
- *canvas:attrColor() -> r, g, b, a : number*
- *canvas:attrFont() -> face : string, size : number, style : string*
- *canvas:attrClip() -> x, y, width, height : number*
  - *Funções utilizadas para obter os valores, respectivamente, das dimensões do canvas, da cor atual, da fonte e da área de corte.*

Funções

- *canvas:attrSize(width, height : number)*
- *canvas:attrColor(r, g, b, a : number)*
- *canvas:attrFont(face : string, size : number, style : string)*
- *canvas:attrClip(x, y, width, height : number)*
  - *Funções utilizadas para modificar os valores, respectivamente, das dimensões do canvas, da cor atual, da fonte e da área de corte.*

Funções

- *canvas:drawLine (x1, y1, x2, y2 : number)*
- *canvas:drawRect (mode : string, x, y, width, height : number)*
- *canvas:drawPolygon (mode : string)*
- *canvas:drawEllipse (mode : string, xc, yc, width, height, ang_start, ang_end : number)*
- *canvas:drawText (text : string, x, y : number)*
  - *Funções utilizadas para desenhar, respectivamente, as primitivas linha, retângulo, polígono, elipse (ou circunferência) e texto.*

Funções

- *canvas:flush()*
  - *Utilizada para enviar o resultado de uma série de operações de desenho e de composição para o canvas, tornando estas visíveis.*
- *canvas:compose (x, y : number, src : canvas, [src_x, src_y, src_w, src_h : number])*
  - Utilizada para copiar o conteúdo de um dado canvas em outro, permitindo operações de composição.

REFERÊNCIAS BIBLIOGRÁFICAS NCL

[1] NETO, Carlos Salles de Soares; SOARES, Luiz Fernando Gomes; RODRIGUES; Rogério Ferreira; BARBOSA, Simone Diniz Junqueira. **Construindo Programas Audiovisuais Interativos Utilizando a NCL 3.0 e a Ferramenta Composer - 2a. edição (NCL 3.0)**, 2007.

[2] Modelo de Contextos Aninhados (Versão 2.2), Laboratório TeleMídia, PUC-RIO, 2003.

[3] DE MELO, Julio César Paulino; ARAÚJO, Rodrigo Moreira. **Os Módulos NCL e NCLUA do Middleware Ginga para aplicações em TV Digital Interativa**. UFRN, 2008.

[4] SOARES, Luiz Fernando Gomes; RODRIGUES; Rogério Ferreira. **Nested Context Model 3.0 Part 1 – NCM Core.** PUC-RIO.

[5] BARBOSA, Simone Diniz Junqueira; SOARES, Luiz Fernando Gomes. **TV Digital Interativa no Brasil se faz com Ginga Fundamentos, Padrões, Autoria Declarativa e Usabilidade**. pp 105-147, PUC-RIO, 2008.

REFERÊNCIAS BIBLIOGRÁFICAS NCL

[6] FILHO, Mauro Fernando de Holanda Beltrão. **GINGAWAY – Uma ferramenta para criação de aplicações GINGA-NCL interativas para TV Digital.** UFPE, 2008.

[7] SANT'ANNA, Francisco; CERQUEIRA, Renato; SOARES, Luiz Fernando Gomes. **NCLUA – Objetos Imperativos LUA na linguagem declarativa NCL.** PUC-RIO.

[8] SANT'ANNA, Francisco; NETO, Carlos de Salles Soares; BARBOSA, Simone Diniz Junqueira; SOARES; Luiz Fernando Gomes. **Aplicações Declarativas NCL com Objetos NCLua Imperativos Embutidos**. PUC-RIO, 2009.

[9] SANT'ANNA, Francisco; CERQUEIRA, Renato; SOARES, Luiz Fernando Gomes. **NCLUA – Objetos Imperativos LUA na linguagem declarativa NCL.** PUC-RIO.

[10] SOARES, Luiz Fernando Gomes; BARBOSA, Simone Diniz Junqueira. **Programando em NCL 3.0 Desenvolvimento de Aplicações para o Middleware Ginga**. 2009.