Capítulo 3

Introducción al Lenguaje NCL

NCL (*Nested Contexto Language*) es un lenguaje declarativo, una aplicación XML, basado en el modelo conceptual NCM. NCL ofrece una clara separación entre los contenidos multimedia y la estructura de una aplicación. Un documento de NCL sólo define como los objetos multimedia son estructurados y relacionados en el tiempo y el espacio. Como lenguaje enlazador, no prescribe ni restringe los tipos de contenido de objetos multimedia en una aplicación.

Este capítulo proporciona una introducción paso a paso de varios elementos que componen el perfil NCL 3.0 para televisión digital. En esta introducción, no hay necesidad de definir cada elemento del lenguaje y cada uno de sus atributos, esto se llevará a cabo en la Parte 2 del libro. La intención es la de proporcionar al lector la capacidad de desarrollar programas NCL sencillos y capacitarlo para comprender y ejercitar los conceptos presentados en la Parte 2. La introducción se realizará a través de un único ejemplo, construido paulatinamente en este capítulo.[1]

[1] Los contenidos de los ejemplos de este capítulo están disponibles en clube.ncl.org.br. "*Programando em NCL 3.0, Desenvolvimentos de aplicações para o Middleware GINGA, Tv Digital e WEB*". Luiz Fernando Gomes Soares, Simone Diniz, Junqueira Barbosa. Traducción a español: Escuela Politécnica del Ejército, ESPE-ECUADOR, Contacto: Gonzalo Olmedo, gfolmedo@espe.edu.ec

3.1 El primer *João*

Para introducirnos en la programación en NCL, vamos utilizar un solo ejemplo, "El primer *João"* (Juan en español), que lo construiremos paso a paso.

El primer *João* se basa en un video, una animación del mismo nombre, producida por André Castellano, que a su vez se basó en las crónicas de Mané Garrincha, escritas por Gerson Soares. La animación cuenta porqué Garrincha pasó a llamar *João* a todos sus marcadores. La historia tiene lugar en una cancha de barrio, donde Mané, un preadolescente pobre, ya hacia su historia. Al ser marcado fuertemente, Garrincha arrasa con dribles desconcertantes al equipo adversario y principalmente a su marcador, el pobre primer *João*, dribles que serían repetidos en la gloriosa carrera del ídolo brasileño.

Vamos a dividir nuestra aplicación en 12 partes. Al final de este capítulo tendremos nuestro programa no-lineal interactivo completo.

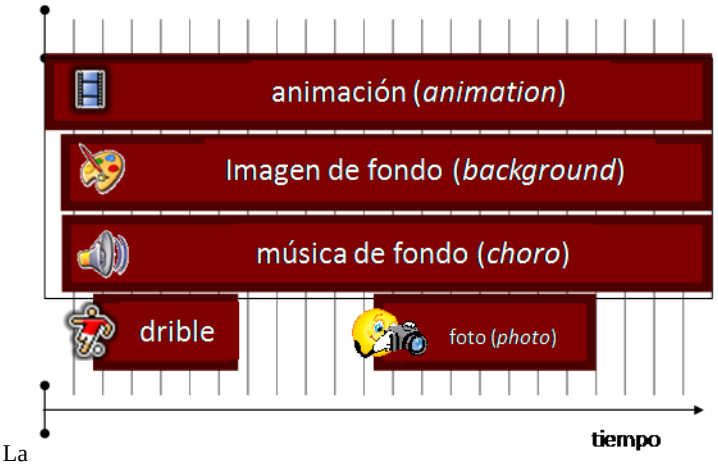
3.2 Sincronismo de multimedia sin interactividad

Durante el video de la animación "El primer *João"*, Garrincha hace su famoso drible en que lleva a los marcadores para la punta, finge que avanza y vuelve, varias veces, hasta que finalmente realiza el drible y pasa. Ese drible fue usado por Mané varias veces en partidos oficiales, inclusive en la Copa del Mundo para la selección brasileña. En otra parte del video, más adelante, aparecen Garrincha y su marcador, caído en el suelo, luego de un drible desconcertante. Una escena similar sucedió en un partido oficial de Botafogo y Vasco, unos años después.

Nuestra primera aplicación, bien simple, muestra lo sencillo que en en NCL introducir varios objetos media sincronizados en el tiempo. Vamos a añadir:

1. una música de fondo llamada "chorinho" (lloro en español), que deberá comenzar después de la presentación inicial del video e iniciar la animación propiamente dicha;
2. una imagen de fondo, sincronizada con el inicio de la animación;
3. otro objeto de video, que se mostrará de forma paralela y sincronizada con el famoso "drible del vaivén" de Mané, representado en la animación; y también
4. otra imagen, una foto, que se mostrará junto con la del marcador en el suelo.

El nuevo vídeo añadido es una réplica, un clon del drible real de Garrincha, realizado por niños de la Comunidad Beira Río, en Río de Janeiro, producido por el Punto de Cultura CIDS-VG, de esa comunidad. Muestra niños descalzos en un campo de tierra, al igual que Garrincha en la época en que aplicó el drible al primer *João*. El mismo Punto de Cultura es también responsable por la foto que muestra exactamente la misma situación que sucede en la animación, cuando el marcador de Garrincha cae al suelo, imagen representada por los mismos niños actores.



La **Figura 3.1** ilustra una visión temporal de la aplicación.

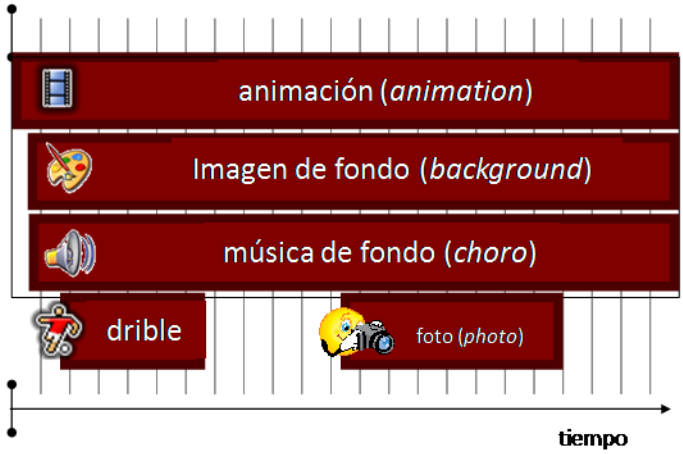


Figura 3.1 Visión temporal.

Para definir todos estos objetos de media, NCL utiliza el elemento <media>, como se ilustra en el Listado 3.1.

```
<media id="background" src="../media/background.png"/>
<media id="animation" src="../media/animGar.mp4"/>
<media id="choro" src="../media/choro.mp3"/>
<media id="drible" src="../media/drible.mp4"/>
<media id="photo" src="../media/photo.png"/>
```

Listado 3.1 Elementos <media>.

Todo elemento <media> tiene un identificador, definido por el atributo *id*, para que pueda ser refernciado más luego. Tiene también un atributo denominado *src*, que contiene un URI para la ubicación del contenido. En todos los casos de nuestro ejemplo, los contenidos están en el directorio "/media", hijo del mismo directorio donde se encuentra la aplicación. Es habitual en aplicaciones para televisión digital, que el atributo *src* para hacer referencia a flujos MPEG. No lo vamos a hacer en nuestro ejemplo, para facilitar su ejecución en los dispositivos más comunes que están disponibles para el lector.

Note que todo elemento NCL debe comenzar con el caracter "<". Cuando un elemento no tiene ningún contenido en sus elementos hijos, debe terminar siempre con "/>". En caso contrario, la definición de los atributos del elemento termina con el carácter ">" y el elemento termina repitiendo su nombre encerrado por los caracteres "</" y ">", al frente y atrás respectivamente, como se ve a continuación en la redefinición del elemento <media id="animation".../>, que en el Listado 3.2 representa el video de animación con dos elementos hijos anidados, denominados <area>.

```
<media id="animation" src="../media/animGar.mp4">
       <area id="segDrible" begin="11.5s"/>
       <area id="segPhoto" begin="41s"/>
</media>
```

Listado 3.2 Elementos <area>.

Los elementos <area> delimitan partes (en tiempo o espacio) del contenido de su objeto de media padre. En este caso, son delimitados el instante en que, en la animación, Garrincha inicia el drible de vaivén y el instante, luego de otro drible, en que el marcador aparece en el piso. El primer instante es delimitado por medio del atributo *begin* igual a 11,5 segundos y el segundo por el atributo *begin* igual a 41 segundos. Como no fue especificado el valor del atributo *end*, se asume por default, que tiene el mismo valor del final del vídeo de la animación en ambos casos. Note que todo elemento <area> también posee un identificador (atributo *id*).

Nuestro siguiente paso en el diseño de la aplicación es definir en qué posición de la pantalla se van a mostrar los diferentes objetos de media. La Figura 3.2 presenta la visión del layout deseada. Se definen regiones para mostrar la figura de fondo ("backgroundReg") y del video de animación

(“screenReg”), ambas en pantalla completa, y una región para ver el video del drible y de la foto (“frameReg”).

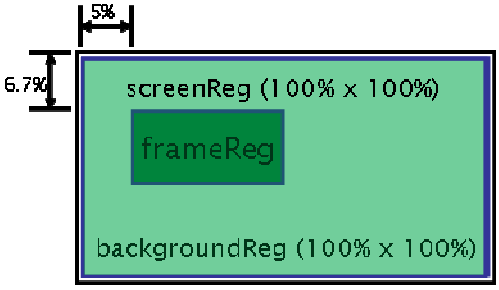


Figura 3.2 Visión de layout.

La definición de los espacios de exhibición puede ser realizada por elementos <property>, hijos de los elementos <media> que representan cada objeto media de la aplicación, como se muestra en el Listado 3.3.

```
<media id="background" src="../media/background.png">
       <property name="width" value="100%"/>
       <property name="height" value="100%"/>
       <property name="zIndex" value="1"/>
</media>
<media id="animation" src="../media/animGar.mp4">
       <area id="segDrible" begin="11.5s"/>
       <area id="segPhoto" begin="41s"/>
       <property name="width" value="100%"/>
       <property name="height" value="100%"/>
       <property name="zIndex" value="2"/>
</media>
<media id="choro" src="../media/choro.mp3"/>
<media id="drible" src="../media/drible.mp4">
       <property name="left" value="5%"/>
       <property name="top" value="6.7%"/>
       <property name="width" value="18.5%"/>
       <property name="height" value="18.5%"/>
       <property name="zIndex" value="3"/>
</media>
<media id="photo" src="../media/photo.png">
       <property name="left" value="5%"/>
       <property name="top" value="6.7%"/>
       <property name="width" value="18.5%"/>
       <property name="height" value="18.5%"/>
       <property name="zIndex" value="3"/>
       <property name="explicitDur" value="5s"/>
</media>
```

Listado 3.3 Elementos <property>.

Todo elemento <property> tiene un identificador válido en el ámbito del objeto media, definido por su atributo name. Por ejemplo “left”, “top”, “widt”, “height”, “right” y “bottom” son propiedades que definen el área de presentación en relación con toda la pantalla del dispositivo de exhibición (valores predeterminados para estos atributos, así como lo que sucede en el caso de valores incoherentes se tratan en la Parte II del libro). Estas propiedades se pueden definir de forma absoluta o como un porcentaje, siempre con respeto a la pantalla completa del dispositivo de visualización.

La propiedad zIndex especifica como las regiones se solapan (se sobreposicionan). Una región de mayor valor para zIndex se sobrepone a otra de menor valor. Así es que en el ejemplo, se define una región de exhibición para ver el video a pantalla completa, y una región de exhibición para ver el video del drible y la foto sobreponiéndose al video mencionado.

Los elementos <property> se pueden utilizar para definir el valor de cualquier propiedad de un objeto de media, no sólo las que definen los espacios de exhibición. A modo de ejemplo, para un objeto de media como una foto, se puede asociar una duración de exhibición de 3 segundos al elemento <media> correspondiente por medio de la propiedad “explicitDur”, como se muestra en el Listado 3.3.

Para finalizar, podemos definir ahora las relaciones de sincronización entre los diferentes objetos de media. En NCL, el elemento <body> agrupa todos los objetos de un documento y sus relaciones.

El elemento <body> es único em um documento NCL y puede tener uno o más elementos <port> como hijos. El elemento <port> indica dónde puede comenzar la exhibición de un documento. En nuestro ejemplo, como se ve en el Listado 3.4, la exhibición comienza siempre por la presentación del video de animación.

Finalmente, podemos ahora definir la relación de sincronización entre los diferentes objetos de multimedia. En NCL, el elemento <body> agrupa todos los objetos de un documento y como ellos se relacionan.

El elemento <body> es único en un documento NCL y puede tener uno o más elementos <port> como hijo. El elemento <port> indica por donde puede comenzar una exhibición del documento. En nuestro ejemplo, como se ilustrada en el Listado 3.4, la exhibición comienza siempre por la presentación del vídeo de la animación.

```
<body>
    <port id="entry" component="animation"/>

    <!—Aquí se incluyen los diferente elementos <media> -->

    <!—Aquí se incluyen los elementos <link> del documento -->
</body>
```

Listado 3.4 Los elementos <body> y <port>.

Las relaciones se definen por elementos <link>, como se ilustra en el Listado 3.5, para nuestro ejemplo del primer *João*. La relación presentada define que, al iniciarse la exhibición del video de animación, el audio con un Chorinho también debe ser iniciado pero 5 segundos después (como determinamos en el enunciado del ejemplo, ese objeto de media es presentado después de los créditos iniciales del video de animación, que duran 5 segundos).

```
<link  id="lMusic" xconnector="onBeginStartDelay">
       <bind role="onBegin" component="animation"/>
       <bind role="start" component="background">
              <bindParam name="delay" value="5s"/>
       </bind>
       <bind role="start" component="choro">
              <bindParam name="delay" value="5s"/>
       </bind>
</link>
```

Listado 3.5 Los elementos <link> y <bind>.

Una relación en NCL se especifica haciendo referencia a la relación por el valor URI del atributo xconnector del elemento <link>, y por los actores que ejercerán los roles de la relación, dados por los elementos <bind>.Un elemento <bind> especifica la función de la relación por medio de su atributo <role>, y la interface que ejecutará el rol, dada por los atributos component, que selecciona un objeto a relacionar; e interface, que selecciona una interface de ese objeto. Por ahora, las únicas interfaces que definimos fueron por medio de los elementos <area>. Cuando no se especifica, el valor por default de la interfaz - interface - asume un área que contiene todo el contenido del objeto.

En la relación definida por el elemento <link> “lMusic” del Listado 3.5, la relación referenciada es definida por el elemento <causalConnector>, hijo del elemento <connectorBase>, como se ilustra en el Listado 3.6. En la relación (Listado 3.5), el rol “onBegin” del conector se asocia al componente “animation”, que, como vimos en el Listado 3.2, se asocia al video de la animación, definiendo que al comenzar la exhibición del video, debe comenzar (*role*=“start”) a la exhibición simultanea de la figura de fondo del audio “chorinho” 5 segundos a partir del video de inicio.

```
<causalConnector id="onBeginStartDelay">
       <connectorParam name="delay"/>
       <simpleCondition role="onBegin"/>
       <simpleAction role="start" delay="$delay" max="unbounded"
                                              qualifier="par"/>
</causalConnector>
```

Listado 3.6 El elemento <cusalConnector> y sus elementos hijos.

La relación es causal, donde la condición es dada por el rol “onBegin”, y la acción es “start”, con el parámetro “delay” (retardo) a ser definido por la relación (el elemento <link>), cuando sea necesario. El atributo *max*=“unbounded” definido en la acción especifica que un número ilimitado de actores puede ejercer este rol. Cuando existe más de un actor para el rol, el atributo *qualifier* define la forma como las acciones deben ser ejecutadas, sea en paralelo o en secuencia.

La Figura 3.3 presenta una visión estructural de la aplicación con todas las relaciones entre los diferentes objetos de media definidos.

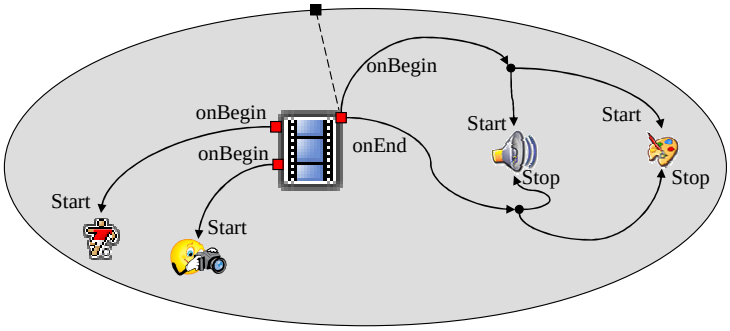


Figura 3.3 Visión estructural.

Las bases de conectores son definidas como elemento hijo del elemento <head> que define, como veremos, las partes reusables de una aplicación. Un mismo conector puede ser usado por más de un elemento <link>. El listado 3.7 presenta la definición completa del elemento <head> de la aplicación ejemplo.

```
<head>
  <connectorBase>
      <causalConnector id="onBeginStartDelay">
        <connectorParam name="delay"/>
        <simpleCondition role="onBegin"/>
        <simpleAction role="start" delay="$delay" max="unbounded"
                                                  qualifier="par"/>
      </causalConnector>

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

      <causalConnector id="onEndStop">
        <simpleCondition role="onEnd"/>
        <simpleAction role="stop" max="unbounded" qualifier="par"/>
      </causalConnector>
   </connectorBase>
</head>
```

Listado 3.7 El elemento <head> y sus elementos hijos.

Ahora podemos definir todo el elemento <body> y sus hijos, para el ejemplo presentado en el Listado 3.8.

```
<body>
     <port id="entry" component="animation"/>
     <media id="background" src="../media/background.png">
       <property name="width" value="100%"/>
       <property name="height" value="100%"/>
       <property name="zIndex" value="1"/>
     </media>
     <media id="animation" src="../media/animGar.mp4">
       <area id="segDrible" begin="11.5s"/>
       <area id="segPhoto" begin="41s"/>
       <property name="width" value="100%"/>
       <property name="height" value="100%"/>
       <property name="zIndex" value="2"/>
     </media>
     <media id="choro" src="../media/choro.mp3"/>
     <media id="drible" src="../media/drible.mp4">
       <property name="left" value="5%"/>
       <property name="top" value="6.7%"/>
       <property name="width" value="18.5%"/>
       <property name="height" value="18.5%"/>
       <property name="zIndex" value="3"/>
     </media>
     <media id="photo" src="../media/photo.png">
       <property name="left" value="5%"/>
       <property name="top" value="6.7%"/>
       <property name="width" value="18.5%"/>
       <property name="height" value="18.5%"/>
       <property name="zIndex" value="3"/>
       <property name="explicitDur" value="5s"/>
     </media>
     <link id="lMusic" xconnector="onBeginStartDelay">
       <bind role="onBegin" component="animation"/>
       <bind role="start" component="background">
         <bindParam name="delay" value="5s"/>
       </bind>
       <bind role="start" component="choro">
         <bindParam name="delay" value="5s"/>
       </bind>
     </link>
     <link id="lDrible" xconnector="onBeginStart">
       <bind role="onBegin" component="animation"
                                              interface="segDrible"/>
       <bind role="start" component="drible"/>
     </link>
     <link id="lPhoto" xconnector="onBeginStart">
       <bind role="onBegin" component="animation"
                                              interface="segPhoto"/>
       <bind role="start" component="photo"/>
     </link>
     <link id="lEnd" xconnector="onEndStop">
       <bind role="onEnd" component="animation"/>
       <bind role="stop" component="choro"/>
       <bind role="stop" component="background"/>
     </link>
```

```
</body>
```

Listado 3.8 El elemento <body> y sus elementos hijos.

El elemento <link> “lDrible” define que al comenzar (*role* “onBegin”) el trecho del video de la animación correspondiente al drible de vaivén de Garrincha (*component* “animation” *interface* “segDrible”), el video del drible debe ser iniciado (*role* “start”).

De forma similar, el elemento <link> “lPhoto” define que al comenzar (*role* “onBegin”), el trecho del video de la animación correspondiente al momento en que el marcador cae en el suelo (*component* “animation” *interface* “segPhoto”), la foto del marcador debe ser exhibida (*role* “start”).

Finalmente, el elemento <link> “lEnd” define que al final (“onEnd”) del video de la animación, la imagen de fondo debe ser terminada (“stop”) y el audio “chorinho” debe parar de tocar.

El lector debe estar intrigado con el hecho de que la imagen de fondo nunca aparece, pues ella quedó todo el tiempo por detrás del vídeo de la animación. En ese caso su definición no tuvo efecto, pero ella tendrá una función a medida que vayamos detallando más nuestro ejemplo.

Los elementos <head> y <body> deben ser declarados como hijos del elemento <ncl>, completando la definición del documento: la aplicación declarativa NCL.

El Listado 3.11 ilustra la aplicación completa. Note que toda aplicación NCL comienza con la línea:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

seguida de la especificación del elemento <ncl>, cuyo atributo *id* define un identificador para la aplicación y el atributo *xmlns* define el espacio de nombres (namespace) donde están definidos los esquemas del perfil NCL EDTV, para verificación, por el analizador XML, de la validez de la aplicación.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Generated by NCL Eclipse -->
<ncl id="sincronismo"
xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
 <head>
    <connectorBase>
      <causalConnector id="onBeginStartDelay">
        <connectorParam name="delay"/>
        <simpleCondition role="onBegin"/>
        <simpleAction role="start" delay="$delay" max="unbounded"
                                                  qualifier="par"/>
      </causalConnector>
      <causalConnector id="onBeginStart">
        <simpleCondition role="onBegin"/>
        <simpleAction role="start" max="unbounded" qualifier="par"/>
      </causalConnector>
      <causalConnector id="onEndStop">
        <simpleCondition role="onEnd"/>
```

```
        <simpleAction role="stop" max="unbounded" qualifier="par"/>
      </causalConnector>
    </connectorBase>
 </head>
 <body>
    <port id="entry" component="animation"/>
    <media id="background" src="../media/background.png">
      <property name="width" value="100%"/>
      <property name="height" value="100%"/>
      <property name="zIndex" value="1"/>
    </media>
    <media id="animation" src="../media/animGar.mp4">
      <area id="segDrible" begin="11.5s"/>
      <area id="segPhoto" begin="41s"/>
      <property name="width" value="100%"/>
      <property name="height" value="100%"/>
      <property name="zIndex" value="2"/>
    </media>
    <media id="choro" src="../media/choro.mp3"/>
    <media id="drible" src="../media/drible.mp4">
      <property name="left" value="5%"/>
      <property name="top" value="6.7%"/>
      <property name="width" value="18.5%"/>
      <property name="height" value="18.5%"/>
      <property name="zIndex" value="3"/>
    </media>
    <media id="photo" src="../media/photo.png">
      <property name="left" value="5%"/>
      <property name="top" value="6.7%"/>
      <property name="width" value="18.5%"/>
      <property name="height" value="18.5%"/>
      <property name="zIndex" value="3"/>
      <property name="explicitDur" value="5s"/>
    </media>

    <link id="lMusic" xconnector="onBeginStartDelay">
      <bind role="onBegin" component="animation"/>
      <bind role="start" component="background">
        <bindParam name="delay" value="5s"/>
      </bind>
      <bind role="start" component="choro">
      <bindParam name="delay" value="5s"/>
      </bind>
   </link>
   <link id="lDrible" xconnector="onBeginStart">
      <bind role="onBegin" component="animation"
                                          interface="segDrible"/>
      <bind role="start" component="drible"/>
   </link>
   <link id="lPhoto" xconnector="onBeginStart">
      <bind role="onBegin" component="animation"
                                          interface="segPhoto"/>
      <bind role="start" component="photo"/>
   </link>
   <link id="lEnd" xconnector="onEndStop">
      <bind role="onEnd" component="animation"/>
      <bind role="stop" component="choro"/>
      <bind role="stop" component="background"/>
   </link>
 </body>
</ncl>
```

Listado 3.11 Documento NCL.

La Figura 3.4 ilustra dos momentos del video de exhibición: el inicio del drible de Garrincha y el inicio de la presentación de la foto, obtenidos durante la ejecución de la aplicación en la implementación de referencia del middleware Ginga-NCL.

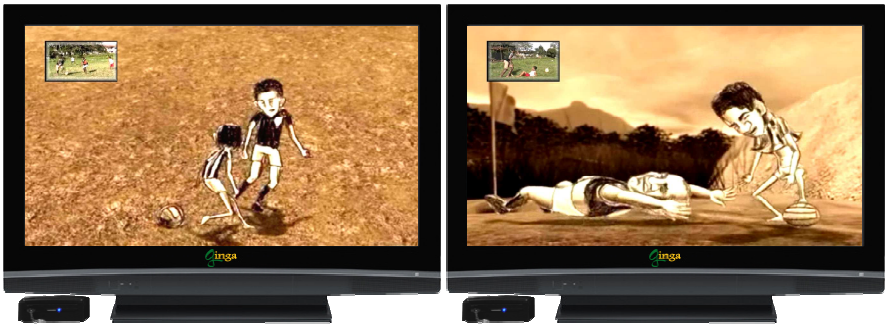


Figura 3.4 Escenas de la aplicación del primer *João*.

3.3 Sincronismo de multimedia sin interactividad, con reúso de características de la presentación e importación de Base de multimedia sin interactividad.

Nuestra segunda aplicación es idéntica a la primera, pero ahora especificada con el reúso de las características de la exhibición de varios objetos de multimedia y con la definición de la base de conectores externa al documento NCL. Es decir, fuera de la nueva sintaxis del documento NCL.

Nuestra primera alteración es respecto a la definición de las regiones donde los diversos objetos de multimedia serán posicionados, los que ya no serán especificados por medio de los elementos <property> hijos de los elementos <media> correspondientes. Para permitir la reutilización de las regiones, ellas serán definidas aparte, por medio de los elementos <region>. El conjunto de elementos <region> es definido como hijo del elemento <regionBase>, como se ilustra en el Listado 3.10.

```
<regionBase>
    <region id="backgroundReg" width="100%" height="100%" zIndex="1"/>
    <region id="screenReg" width="100%" height="100%" zIndex="2">
         <region id="frameReg" left="5%" top="6.7%" width="18.5%"
                                    height="18.5%" zIndex="3"/>
    </region>
</regionBase>
```

Listado 3.10 Elementos <region> y <regionBase>.

Todo elemento <region> posee un identificador y los atributos *left, top width, height, right y zIndex*, que definen su área de exhibición en relación a la región padre (los valores por *default* para esos atributos, así como lo que sucede en caso de inconsistencia, son discutidos en la segunda parte del libro). Estos atributos pueden ser definidos de forma absoluta o como porcentaje, pero ahora siempre con relación a los mismos atributos de la región del padre. Cuando el elemento no posee una región padre, la posición es realizada con relación a la pantalla total del dispositivo de exhibición. El atributo *zIndex* específica como queda la superposición de las regiones. Por ejemplo, en el Listado 3.10, la región "frameReg" es definida como hija de la región "screenReg", es decir, todos los valores definidos para su posición son relativos a la región padre "frameReg". Adicionalmente, se debe notar que esta región será reusada tanto para la definición de la posición de la foto, como también para la posición del video del drible.

Avanzando en nuestro proyecto de aplicación, vamos ahora a especificar como las regiones definidas (elementos <region>) son asociadas a los diferentes objetos de multimedia definidos (elementos <media>). Eso se da a través de los elementos <descriptor>. El conjunto de todos los descriptores es definido como hijo del elemento <descriptorBase>, como se ilustra en el Listado 3.11.

```
<descriptorBase>
    <descriptor id="backgroundDesc" region="backgroundReg"/>
    <descriptor id="screenDesc" region="screenReg"/>
    <descriptor id="photoDesc" region="frameReg" explicitDur="5s"/>
    <descriptor id="audioDesc"/>
    <descriptor id="dribleDesc" region="frameReg"/>
</descriptorBase>
```

Listado 3.11 Elementos <descriptor> y <descriptorBase>.

El elemento <descriptor> especifica todas las características iníciales para la exhibición de un objeto de multimedia, inclusive su posición, dada por su atributo *region*, que es usado para referenciar un elemento <region>. Un descriptor puede tener otros atributos adicionales, como el atributo *explicitDur*, que determina el tiempo de duración explícita de un objeto multimedia. Por ejemplo, el elemento <descriptor> "photoDesc" especifica que el objeto multimedia debe ser exhibido por 5 segundos. Las propiedades definidas por el elemento <descriptor> pueden ser reusadas en la definición de varios objetos multimedia.

Es muy importante resaltar que los elementos <region> y <descriptor> no definen nuevas propiedades, lo que definen son valores iníciales para propiedades reservadas del objeto multimedia. Varias de estas propiedades serán presentadas a medida que avancemos el ejemplo. Una lista de estas propiedades es presentada en el Capítulo 9.

Como mencionamos, un elemento <descriptor> es conectado a un objeto de multimedia por el atributo *descriptor* del elemento <media> que

representa el objeto, como se ilustra en el Código del Programa 3.12, en la definición de todos los elementos <media> ahora sin los elementos <property> hijos.

```
<media id="background" src="../media/background.png"
                                        descriptor="backgroundDesc"/>
<media id="animation" src="../media/animGar.mp4"
descriptor="screenDesc">
  <area id="segDrible" begin="12s"/>
  <area id="segPhoto" begin="41s"/>
</media>
<media id="choro" src="../media/choro.mp3" descriptor="audioDesc"/>
<media id="drible" src="../media/drible.mp4" descriptor="dribleDesc"/>
<media id="photo" src="../media/photo.png" descriptor="photoDesc"/>
```

Listado 3.12 Elementos <media> redefinidos.

Pongamos nuestra atención ahora en los conectores. En las relaciones definidas por los elementos <link>, la relación referenciada puede ser definida en un documento externo, que debe ser importado para el documento en cuestión. Eso es especificado por el elemento <importBase>, hijo del elemento <connectorBase>, conforme se ilustra el Listado 3.13. En el Listado, los conectores están definidos en un archivo con el nombre "causalConnBase.ncl", en el mismo directorio donde se encuentra nuestro documento NCL de ejemplo. Este archivo de conectores será usado en los ejemplos siguientes con el alias "conEx", definido por el atributo *alias*. El contenido de este archivo es presentado en el Apéndice D.

```
<connectorBase>
    <importBase documentURI="../causalConnBase.ncl" alias="conEx"/>
</connectorBase>
```

Listado 3.13 Elementos <importBase> y <connectorBase>.

Al referenciar un conector definido externamente al documento, un elemento <link> debe concatenar el valor del atributo alias con el carácter especial #, seguido por el identificador del conector. Los mismos conectores definidos en la Sección 3.2, al ser definidos externamente en el archivo "causalConnBase.ncl", deben ser referenciados como ilustra el Listado 3.14, que presenta la redefinición de todos los elementos <link> del ejemplo del primer *João*.

```
<body>
   <link id="lMusic" xconnector="conEx#onBeginStartDelay">
     <bind role="onBegin" component="animation"/>
     <bind role="start" component="background">
       <bindParam name="delay" value="5s"/>
     </bind>
     <bind role="start" component="choro">
       <bindParam name="delay" value="5s"/>
     </bind>
  </link>
  <link  id="lDrible" xconnector="conEx#onBeginStart">
     <bind role="onBegin" component="animation"
```

```
interface="segDrible"/>
     <bind role="start" component="drible"/>
  </link>
  <link id="lPhoto" xconnector="conEx#onBeginStart">
     <bind role="onBegin" component="animation" interface="segPhoto"/>
     <bind role="start" component="photo"/>
  </link>
  <link id="lEnd" xconnector="conEx#onEndStop">
     <bind role="onEnd" component="animation"/>
     <bind role="stop" component="choro"/>
     <bind role="stop" component="background"/>
  </link>
 </body>
```

Listado 3.14 Redefinición de los elementos <link>.

Todas las bases que definimos, <regionBase>, <descriptorBase> y <connectorBase>, deben ser hijas del elemento <head> del documento NCL, como se ilustra en el Listado 3.15.

```
<head>
   <regionBase>
    <region id="backgroundReg" width="100%" height="100%" zIndex="1"/>
    <region id="screenReg" width="100%" height="100%" zIndex="2">
    <region id="frameReg" left="5%" top="6.7%" width="18.5%"
                                      height="18.5%" zIndex="3"/>
      </region>
   </regionBase>
   <descriptorBase>
    <descriptor id="backgroundDesc" region="backgroundReg"/>
    <descriptor id="screenDesc" region="screenReg"/>
    <descriptor id="photoDesc" region="frameReg" explicitDur="5s"/>
    <descriptor id="audioDesc"/>
    <descriptor id="dribleDesc" region="frameReg"/>
   </descriptorBase>
   <connectorBase>
    <importBase documentURI="../causalConnBase.ncl" alias="conEx"/>
   </connectorBase>
</head>
```

Listado 3.15 Elemento <head> y sus elementos hijos.

El Listado 3.16 presenta la aplicación completa.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Ejemplo de sincronismo sin la interacción del usuario, con reúso de las
características de exhibición e importación de base -->
<ncl id="sync" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
 <head>
   <regionBase>
    <region id="backgroundReg" width="100%" height="100%" zIndex="1"/>
    <region id="screenReg" width="100%" height="100%" zIndex="2">
    <region id="frameReg" left="5%" top="6.7%" width="18.5%" height="18.5%"
                                                          zIndex="3"/>
      </region>
   </regionBase>
   <descriptorBase>
    <descriptor id="backgroundDesc" region="backgroundReg"/>
    <descriptor id="screenDesc" region="screenReg"/>
    <descriptor id="photoDesc" region="frameReg" explicitDur="5s"/>
    <descriptor id="audioDesc"/>
    <descriptor id="dribleDesc" region="frameReg"/>
   </descriptorBase>
```

```
   <connectorBase>
    <importBase documentURI="../causalConnBase.ncl" alias="conEx"/>
   </connectorBase>
 </head>
 <body>
   <port id="entry" component="animation"/>
   <media id="background" src="../media/background.png"
                                              descriptor="backgroundDesc"/>
   <media id="animation" src="../media/animGar.mp4" descriptor="screenDesc">
    <area id="segDrible" begin="12s"/>
    <area id="segPhoto" begin="41s"/>
   </media>
   <media id="choro" src="../media/choro.mp3" descriptor="audioDesc"/>
   <media id="drible" src="../media/drible.mp4" descriptor="dribleDesc"/>
   <media id="photo" src="../media/photo.png" descriptor="photoDesc"/>
   <link id="lMusic" xconnector="conEx#onBeginStartDelay">
     <bind role="onBegin" component="animation"/>
     <bind role="start" component="background">
       <bindParam name="delay" value="5s"/>
     </bind>
     <bind role="start" component="choro">
       <bindParam name="delay" value="5s"/>
     </bind>
  </link>
  <link  id="lDrible" xconnector="conEx#onBeginStart">
     <bind role="onBegin" component="animation" interface="segDrible"/>
     <bind role="start" component="drible"/>
  </link>
  <link id="lPhoto" xconnector="conEx#onBeginStart">
     <bind role="onBegin" component="animation" interface="segPhoto"/>
     <bind role="start" component="photo"/>
  </link>
  <link id="lEnd" xconnector="conEx#onEndStop">
     <bind role="onEnd" component="animation"/>
     <bind role="stop" component="choro"/>
     <bind role="stop" component="background"/>
  </link>
 </body>
 </ncl>
```

Listado 3.16 Documento NCL con reutilización e importación.

3.4 Adicionando sincronismo con interactividad

Ahora vamos a incrementar nuestra aplicación, introduciendo un ejemplo de sincronismo temporal con la interacción del usuario.

En un cierto trecho del video de la animación, al caer en el piso el adversario de nuestro Garrincha muestra su zapato. En este instante, haremos aparecer el ícono de un zapato, que al ser seleccionado por el telespectador a través del botón rojo del control remoto, hará aparecer un video de propaganda del zapato. Recordando que en la animación, Garrincha era un niño pobre, él jugaba descalzo. De esta forma, en el video de la propaganda, también realizado por el Punto de Cultura CIDS-VG, aparece un niño, el mismo que representó a Garrincha en el video del drible y en la foto, soñando en poseer un zapato de fútbol. Durante la exhibición del video de la propaganda, el video de la animación deberá ser redimensionado, posibilitando la exhibición simultánea de los dos videos sin superposición.

La Figura 3.5 ilustra la visión estructural de la nueva versión de la aplicación. Note que fueron incrementaron dos objetos de multimedia, de un nuevo vínculo para el video de animación y tres nuevas relaciones.

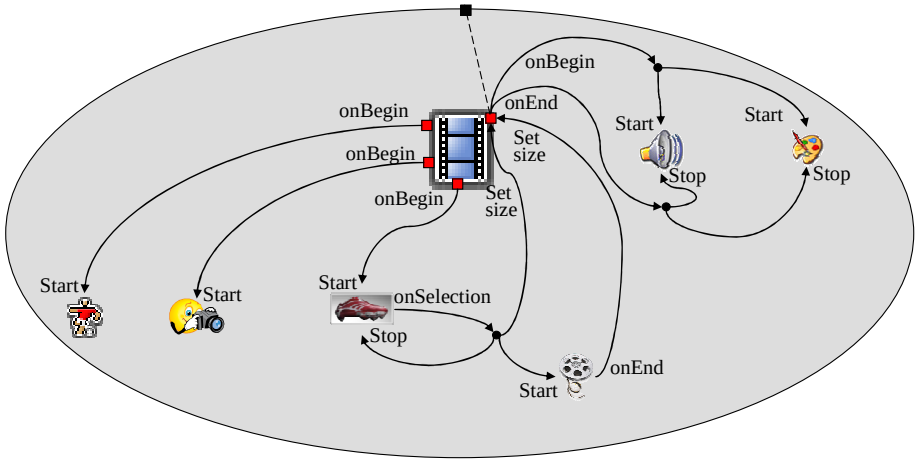


Figura 3.5 Visión estructural.

Tenemos que modificar la definición del elemento <media> que representa la animación, introduciendo un nuevo vínculo, es decir, un nuevo elemento <area>, correspondiendo al tiempo en que el ícono del zapato deberá aparecer. Como el video de la animación debe ser redimensionado, caso el ícono sea seleccionado, es necesario definir las propiedades de la visualización (*layout*) que serán modificadas. Las propiedades de un objeto de multimedia susceptibles de manipulación a través de relaciones deben ser definidas por medio de elementos <property>, así sus valores iníciales, hayan sido definidos usando elementos <descriptor> o <region>. Para nuestro caso, queremos manipular las variables de posición y dimensión ("*left*", "*top*", "*width*" y "*height*"), representadas por el valor "*bounds*", como se ilustra en el Listado 3.17, donde también es ilustrada la inclusión de los dos nuevos objetos de multimedia en el conjunto de los objetos de multimedia anteriormente definido: el ícono ("icon") y el video de propaganda ("shoes").

```
<media id="background" src="../media/background.png"
                                                descriptor="backgroundDesc"/>
<media id="animation" src="../media/animGar.mp4" descriptor="screenDesc">
      <area id="segDrible" begin="12s"/>
      <area id="segPhoto" begin="41s"/>
      <area id="segIcon" begin="45s" end="51s"/>
      <property name="bounds"/>
</media>
<media id="choro" src="../media/choro.mp3" descriptor="audioDesc"/>
<media id="drible" src="../media/drible.mp4" descriptor="dribleDesc"/>
<media id="photo" src="../media/photo.png" descriptor="photoDesc"/>
<media id="icon" src="../media/icon.png" descriptor="iconDesc"/>
<media id="shoes" src="../media/shoes.mp4" descriptor="shoesDesc"/>
```

Listado 3.17 Conjunto de elementos <media> de la nueva versión de la aplicación.

Tenemos ahora que incrementar los descriptores y las regiones para la presentación de los nuevos objetos de multimedia, como se ilustra en el Listado 3.18. Note que las nuevas regiones fueron definidas como hijas del elemento <region> “screenReg”, es decir, relativas a esta región previamente definida.

```
<regionBase>
   <region id="backgroundReg" width="100%" height="100%" zIndex="1"/>
   <region id="screenReg" width="100%" height="100%" zIndex="2">
      <region id="frameReg" left="5%" top="6.7%" width="18.5%"
                                                 height="18.5%" zIndex="3"/>
      <region id="iconReg" left="87.5%" top="11.7%"
                                  width="8.45%" height="6.7%" zIndex="3"/>
      <region id="shoesReg" left="15%" top="60%" width="25%"
                                                  height="25%" zIndex="3"/>
   </region>
   </regionBase>

<descriptorBase>
   <descriptor id="backgroundDesc" region="backgroundReg"/>
   <descriptor id="screenDesc" region="screenReg"/>
   <descriptor id="photoDesc" region="frameReg"explicitDur="5s"/>
   <descriptor id="audioDesc"/>
   <descriptor id="dribleDesc" region="frameReg"/>
   <descriptor id="iconDesc" region="iconReg" e xplicitDur="6s"/>
   <descriptor id="shoesDesc" region="shoesReg"/>
</descriptorBase>
```

Listado 3.18 Definición de nuevos elementos <region> y de los nuevos elementos <descriptor>.

Ahora solo resta definir las tres nuevas relaciones introducidas, como se ilustra en el Listado 3.19.

```
<link  id="lIcon" xconnector="conEx#onBeginStart">
       <bind role="onBegin" component="animation"
interface="segIcon"/>
        <bind role="start" component="icon"/>
</link>
<link  id="lAdvert" xconnector="conEx#onKeySelectionStopSetStart">
       <bind role="onSelection" component="icon">
          <bindParam name="keyCode" value="RED"/>
       </bind>
       <bind role="set" component="animation" interface="bounds">
          <bindParam name="varSet" value="5%,6.67%,45%,45%"/>
       </bind>
       <bind role="start" component="shoes"/>
       <bind role="stop" component="icon"/>
</link>
<link id="lEndAdvert" xconnector="conEx#onEndSet">
       <bind role="onEnd" component="shoes"/>
       <bind role="set" component="animation" interface="bounds">
          <bindParam name="varSet" value="0,0,222.22%,222.22%"/>
       </bind>
</link>
```

Listado 3.19 Definición de las nuevas relaciones.

El primero elemento <link> (“lIcon”) define que al comenzar la presentación de la trama de la animación con la condición “onBegin”, aparecerá el ícono con la acción “start”.

El segundo elemento <link> (“lAdvert”) es más complejo. Define que si el ícono del zapato fuera seleccionado con la tecla roja del control remoto con la condición “onSelection”, pasada como parámetro correspondiente, *keycode*=“RED”, el video de la animación debe ser redimensionado (acción “set” en la propiedad “bounds” definida en el elemento “animation”) para los valores pasados como parámetros (*left*=“5%”, *top*=“6,67%”, *width* y *height* =“45%”, todos relativos a los valores iníciales). La relación también determina que la propaganda del zapato debe ser iniciada (acción “start” en el elemento “shoes”) y que el ícono debe parar su exhibición (acción “stop” en el elemento “icon”).

El tercer elemento <link> (“lEndAdvert”) especifica que cuando el video correspondiente a la propaganda del zapato termina (condición “onEnd”), el video de la animación debe retomar sus dimensiones originales (acción “set” en la propiedad “bounds” definida en el elemento “animation”).

El Listado 3.20 presenta el programa NCL correspondiente a la nueva versión de la aplicación.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!--Ejemplo de sincronismo e interactividad-->
<ncl id="syncInt" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
 <head>
   <regionBase>
     <region id="backgroundReg" width="100%" height="100%" zIndex="1"/>
     <region id="screenReg" width="100%" height="100%" zIndex="2">
      <region id="frameReg" left="5%" top="6.7%" width="18.5%"
                                          height="18.5%" zIndex="3"/>
      <region id="iconReg" left="87.5%" top="11.7%"
                                 width="8.45%" height="6.7%" zIndex="3"/>
      <region id="shoesReg" left="15%" top="60%" width="25%"
                                          height="25%" zIndex="3"/>
     </region>
   </regionBase>
   <descriptorBase>
     <descriptor id="backgroundDesc" region="backgroundReg"/>
     <descriptor id="screenDesc" region="screenReg"/>
     <descriptor id="photoDesc" region="frameReg"explicitDur="5s"/>
     <descriptor id="audioDesc"/>
     <descriptor id="dribleDesc" region="frameReg"/>
     <descriptor id="iconDesc" region="iconReg" e xplicitDur="6s"/>
     <descriptor id="shoesDesc" region="shoesReg"/>
   </descriptorBase>

   <connectorBase>
        <importBase documentURI="../causalConnBase.ncl" alias="conEx"/>
      </connectorBase>
 </head>

 <body>
    <port id="entry" component="animation"/>
    <media id="background" src="../media/background.png"
                                                descriptor="backgroundDesc"/>
    <media id="animation" src="../media/animGar.mp4" descriptor="screenDesc">
      <area id="segDrible" begin="12s"/>
```

```
      <area id="segPhoto" begin="41s"/>
      <area id="segIcon" begin="45s" end="51s"/>
         <property name="bounds"/>
    </media>

    <media id="choro" src="../media/choro.mp3" descriptor="audioDesc"/>
    <media id="drible" src="../media/drible.mp4" descriptor="dribleDesc"/>
    <media id="photo" src="../media/photo.png" descriptor="photoDesc"/>
    <media id="icon" src="../media/icon.png" descriptor="iconDesc"/>
    <media id="shoes" src="../media/shoes.mp4" descriptor="shoesDesc"/>

    <link id="lMusic" xconnector="conEx#onBeginStartDelay">
         <bind role="onBegin" component="animation"/>
         <bind role="start" component="background">
            <bindParam name="delay" value="5s"/>
         </bind>
         <bind role="start" component="choro">
            <bindParam name="delay" value="5s"/>
         </bind>
    </link>
    <link id="lDrible" xconnector="conEx#onBeginStart">
         <bind role="onBegin" component="animation" interface="segDrible"/>
         <bind role="start" component="drible"/>
    </link>
    <link  id="lPhoto" xconnector="conEx#onBeginStart">
         <bind role="onBegin" component="animation" interface="segPhoto"/>
         <bind role="start" component="photo"/>
    </link>
    <link  id="lEnd" xconnector="conEx#onEndStop">
         <bind role="onEnd" component="animation"/>
         <bind role="stop" component="background"/>
         <bind role="stop" component="choro"/>
    </link>
    <link  id="lIcon" xconnector="conEx#onBeginStart">
         <bind role="onBegin" component="animation" interface="segIcon"/>
         <bind role="start" component="icon"/>
    </link>
    <link  id="lAdvert" xconnector="conEx#onKeySelectionStopSetStart">
         <bind role="onSelection" component="icon">
            <bindParam name="keyCode" value="RED"/>
         </bind>
         <bind role="set" component="animation" interface="bounds">
            <bindParam name="varSet" value="5%,6.67%,45%,45%"/>
         </bind>
         <bind role="start" component="shoes"/>
         <bind role="stop" component="icon"/>
     </link>
     <link  id="lEndAdvert" xconnector="conEx#onEndSet">
         <bind role="onEnd" component="shoes"/>
         <bind role="set" component="animation" interface="bounds">
            <bindParam name="varSet" value="0,0,222.22%,222.22%"/>
          </bind>
     </link>
   </body>
</ncl>
```

Listado 3.20 El primer *João* con sincronismo e interactividad.

La Figura 3.6 ilustra dos momentos da exhibición, el inicio cuando aparece el ícono y el momento de la presentación de la propaganda, obtenidos durante la ejecución de la aplicación en la implementación de referencia del middleware Ginga-NCL.

Figura 3.6 Escenas de la aplicación del primer *João* con interactividad.

3.5 Adicionando el uso de contextos

En NCL, los elementos <context> pueden ser usados para estructurar una aplicación. En esta sección vamos a ejemplificar el uso de contextos, agrupando todos los elementos de la propaganda del zapato. Esto posibilitaría, además de mayor estructuración del programa, la reutilización del código de toda la estructura. Vamos a tratar la reutilización en la próxima sección.

La Figura 3.7 ilustra la visión estructural de la nueva versión de la aplicación.

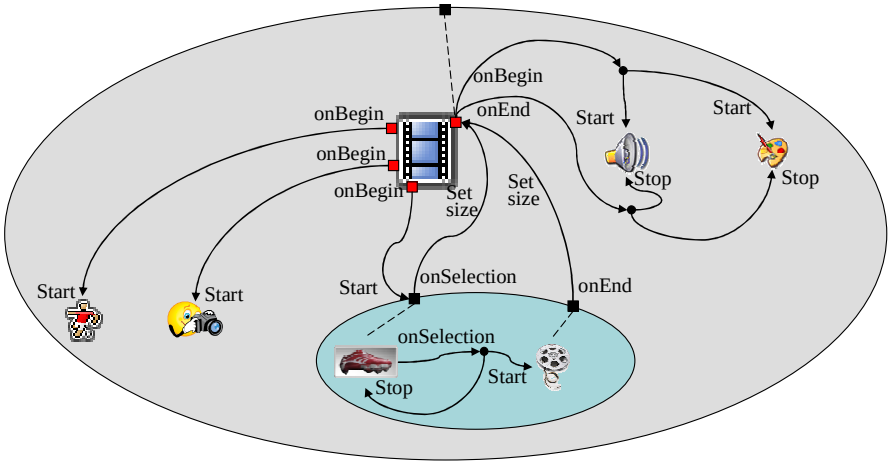


Figura 3.7 Escenas de la aplicación el primer *João* utilizando contextos.

Se puede observar en la Figura 3.7, que no es posible acceder directamente objetos de multimedia dentro de un contexto. Como se vio en el Capítulo 2, el acceso debe ser realizado a través de los elementos <port>, que son responsables, por exportar lo que es visible dentro de un contexto usando relaciones externas. Por lo tanto, debemos no solo incluir los

elementos <media> “icon” y “shoes” dentro de un elemento <context>, sino también redefinir las tres relaciones definidas en la sección anterior. Note que una de las relaciones puede ser definida dentro del propio contexto, una vez que solo involucra elementos que son sus hijos. De esta forma, la definición del contexto queda como se ilustra en el Listado 3.21.

```
<context id="advert">
 <port id="pIcon" component="icon"/>
 <port id="pShoes" component="shoes"/>
 <media id="icon" src="../media/icon.png" descriptor="iconDesc"/>
 <media id="shoes" src="../media/shoes.mp4" descriptor="shoesDesc"/>

 <link id="lBeginShoes" xconnector="conEx#onKeySelectionStopStart">
   <bind role="onSelection" component="icon">
      <bindParam name="keyCode" value="RED"/>
   </bind>
   <bind role="start" component="shoes"/>
   <bind role="stop" component="icon"/>
 </link>
</context>
```

Listado 3.21 Uso del elemento <context> en la definición de la propaganda del zapato.

Observe que el contexto ofrece dos elementos <port> para el mundo exterior: uno para su componente “icon”, que será usado como entrada en el comando de inicio de la presentación del ícono y como salida en el comando para el redimensionamiento del video de la animación; y otro para el componente “shoes”, que será utilizado como salida en el comando para restaurar el video de la animación en sus dimensiones originales.

Las tres relaciones definidas en la sección anterior fueron disgregadas en cuatro, para que una relación sea definida dentro del contexto. Esta relación, identificada como “lBeginShoes” en el Listado 3.21, especifica que al ser seleccionado (condición “onSelection”) el ícono del zapato, por medio del botón rojo del control remoto, la propaganda debe iniciar (acción “start” sobre el componente “shoes”) y la figura del ícono debe finalizar su exhibición (acción “stop” sobre el componente “icon”).

Las tres relaciones restantes son especificadas como hijas del elemento <body>, substituyendo las tres relaciones definidas en la sección anterior, como se ilustra en el Listado 3.22, que presenta el nuevo programa para esta nueva versión. Observe las nuevas relaciones “lIcon”, “lAdvert”, “lEndAdvert”.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Ejemplo de uso de contexto -->
<ncl id="contexto" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
 <head>
  <regionBase>
    <region id="backgroundReg" width="100%" height="100%" zIndex="1"/>
    <region id="screenReg" width="100%" height="100%" zIndex="2">
      <region id="frameReg" left="5%" top="6.7%" width="18.5%"
                                          height="18.5%" zIndex="3"/>
      <region id="iconReg" left="87.5%" top="11.7%" width="8.45%"
                                          height="6.7%" zIndex="3"/>
      <region id="shoesReg" left="15%" top="60%" width="25%"
```

```
                                        height="25%" zIndex="3"/>
    </region>
  </regionBase>
  <descriptorBase>
    <descriptor id="backgroundDesc" region="backgroundReg"/>
    <descriptor id="screenDesc" region="screenReg"/>
    <descriptor id="photoDesc" region="frameReg" explicitDur="5s">
      <descriptorParam name="transparency" value="0.6"/>
    </descriptor>
    <descriptor id="audioDesc"/>
    <descriptor id="dribleDesc" region="frameReg"/>
    <descriptor id="iconDesc" region="iconReg" explicitDur="6s"/>
    <descriptor id="shoesDesc" region="shoesReg"/>
  </descriptorBase>
  <connectorBase>
    <importBase documentURI="../causalConnBase.ncl" alias="conEx"/>
  </connectorBase>
 </head>

 <body>
   <port id="entry" component="animation"/>
   <media id="background" src="../media/background.png"
                                        descriptor="backgroundDesc"/>
   <media id="animation" src="../media/animGar.mp4"
                                        descriptor="screenDesc">
          <area id="segDrible" begin="12s"/>
          <area id="segPhoto" begin="41s"/>
          <area id="segIcon" begin="45s" end="51s"/>
          <property name="bounds"/>
   </media>
   <media id="choro" src="../media/choro.mp3"
                                        descriptor="audioDesc"/>
   <media id="drible" src="../media/drible.mp4"
                                        descriptor="dribleDesc"/>
   <media id="photo" src="../media/photo.png"
                                        descriptor="photoDesc"/>

   <context id="advert">
    <port id="pIcon" component="icon"/>
    <port id="pShoes" component="shoes"/>
    <media id="icon" src="../media/icon.png" descriptor="iconDesc"/>
    <media id="shoes" src="../media/shoes.mp4" descriptor="shoesDesc"/>

    <link id="lBeginShoes" xconnector="conEx#onKeySelectionStopStart">
       <bind role="onSelection" component="icon">
          <bindParam name="keyCode" value="RED"/>
       </bind>
       <bind role="start" component="shoes"/>
          <bind role="stop" component="icon"/>
    </link>
   </context>

   <link id="lMusic" xconnector="conEx#onBeginStartDelay">
       <bind role="onBegin" component="animation"/>
       <bind role="start" component="background">
          <bindParam name="delay" value="5s"/>
       </bind>
       <bind role="start" component="choro">
          <bindParam name="delay" value="5s"/>
       </bind>
   </link>
   <link id="lDrible" xconnector="conEx#onBeginStart">
       <bind role="onBegin" component="animation" interface="segDrible"/>
       <bind role="start" component="drible"/>
   </link>
   <link id="lPhoto" xconnector="conEx#onBeginStart">
       <bind role="onBegin" component="animation" interface="segPhoto"/>
       <bind role="start" component="photo"/>
   </link>
   <link id="lEnd" xconnector="conEx#onEndStop">
```

```
      <bind role="onEnd" component="animation"/>
      <bind role="stop" component="background"/>
      <bind role="stop" component="choro"/>
   </link>
   <link id="lIcon" xconnector="conEx#onBeginStart">
      <bind role="onBegin" component="animation" interface="segIcon"/>
      <bind role="start" component="advert" interface="pIcon"/>
   </link>
   <link id="lAdvert" xconnector="conEx#onKeySelectionSet">
      <bind role="onSelection" component="advert" interface="pIcon">
        <bindParam name="keyCode" value="RED"/>
      </bind>
      <bind role="set" component="animation" interface="bounds">
        <bindParam name="varSet" value="5%,6.67%,45%,45%"/>
      </bind>
   </link>
   <link id="lEndAdvert" xconnector="conEx#onEndSet">
      <bind role="onEnd" component="advert" interface="pShoes"/>
      <bind role="set" component="animation" interface="bounds">
         <bindParam name="varSet" value="0,0,222.22%,222.22%"/>
      </bind>
   </link>
  </body>
</ncl>
```

Listado 3.22 El primer *João* utilizando contextos.

3.6 Adicionando reúso de objetos multimedia

El reúso de objetos multimedia y de contexto torna una aplicación más limpia, más fácil de mantener y menos sujeta a errores. Mismo, corriendo el riesgo de tornar el contexto definido en la sección anterior menos susceptible de reúso, en otras partes del video de la animación (o mismo en otro video), vamos utilizar ese contexto para introducir la reutilización de objetos multimedia, apenas con el objetivo de tornar más fácil la definición de relaciones.

Lo que haremos es crear un nuevo objeto multimedia dentro del elemento <context> “advert”, que heredará toda la definición del elemento <media> “animation” y que además, representará el mismo objeto multimedia: el video de la animación. Como fue discutido en la sección 2, el modelo NCL no permite que un mismo objeto esté contenido en más de un contexto, pero esto puede realizarse a través del reúso.

La Figura 3.8 ilustra el reúso del elemento <media> “animation”. Note la simplificación de la definición de los enlaces vinculados a la propaganda de los zapatos, ya que ahora son internos al contexto y no necesitan más utilizar puertos para el acceso al video de la animación.

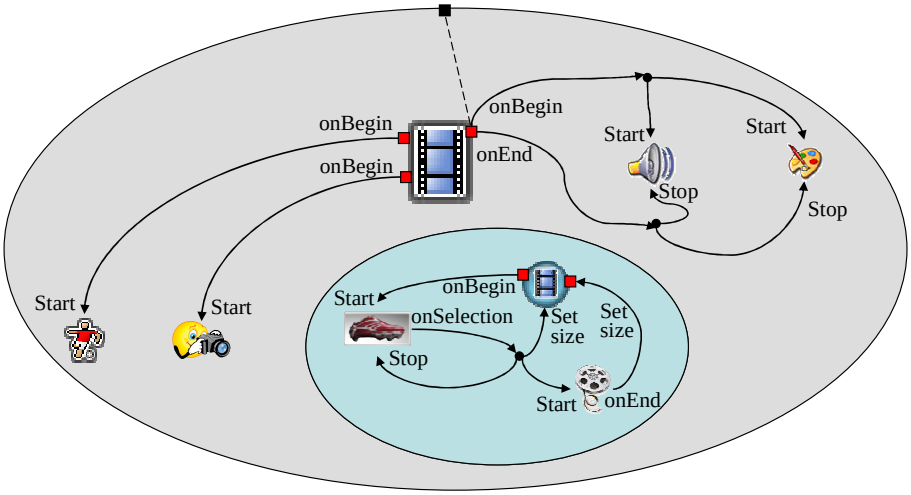


Figura 3.8 Reúso.

El nuevo elemento <media> definido dentro del contexto tiene la definición dada por el Listado 3.23. Note que el uso del atributo *refer* indica que el elemento heredará toda la definición del elemento referido, en este el caso “animation”. El elemento referenciado y el elemento que lo referencia deben ser considerados el mismo nodo en relación a la presentación, si el atributo *instance* recibe un valor “instSame”, como en el presente caso.

```
<media id="reusedAnimation" refer="animation" instance="instSame">
  <property name="bounds"/>
</media>
```

Listado 3.23 Reúso de un objeto de multimedia.

Note también que en el nuevo elemento, nuevas interfaces (elementos <area> y <property>) pueden ser creadas. En este caso, para ejemplificar, retiraremos la definición de la propiedad “bounds” del elemento “animation” y la incluiremos en el reúso del elemento, como es indicado en el Listado 3.23.

En esta nueva versión, las cuatro relaciones introducidas en la sección anterior deben ser modificadas y redefinidas como si pertenecieran al contexto. Las cuatro relaciones vuelven a ser apenas tres, y ahora deben referenciar el elemento <media> “reusedAnimation”, como ilustra el Listado 3.24, conteniendo todo el nuevo programa NCL.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Ejemplo de Reúso -->
<ncl id="reuse" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
  <head>
    <regionBase>
      <region id="backgroundReg" width="100%" height="100%" zIndex="1"/>
      <region id="screenReg" width="100%" height="100%" zIndex="2">
```

```
      <region id="frameReg" left="5%" top="6.7%" width="18.5%" height="18.5%"
                                                        zIndex="3"/>
      <region id="iconReg" left="87.5%" top="11.7%" width="8.45%" height="6.7%"
                                                        zIndex="3"/>
      <region id="shoesReg" left="15%" top="60%" width="25%" height="25%"
                                                        zIndex="3"/>
      </region>
  </regionBase>
  <descriptorBase>
    <descriptor id="backgroundDesc" region="backgroundReg"/>
    <descriptor id="screenDesc" region="screenReg"/>
    <descriptor id="photoDesc" region="frameReg" explicitDur="5s"/>
    <descriptor id="audioDesc"/>
    <descriptor id="dribleDesc" region="frameReg"/>
    <descriptor id="iconDesc" region="iconReg" explicitDur="6s"/>
    <descriptor id="shoesDesc" region="shoesReg"/>
  </descriptorBase>
  <connectorBase>
    <importBase documentURI="../causalConnBase.ncl" alias="conEx"/>
  </connectorBase>
</head>

<body>
  <port id="entry" component="animation"/>
  <media id="background" src="../media/background.png"
                                    descriptor="backgroundDesc"/>
    <media id="animation" src="../media/animGar.mp4" descriptor="screenDesc">
      <area id="segDrible" begin="12s"/>
      <area id="segPhoto" begin="41s"/>
      <area id="segIcon" begin="45s" end="51s"/>
    </media>
    <media id="choro" src="../media/choro.mp3" descriptor="audioDesc"/>
    <media id="drible" src="../media/drible.mp4" descriptor="dribleDesc"/>
    <media id="photo" src="../media/photo.png" descriptor="photoDesc"/>
    <context id="advert">
     <media id="reusedAnimation" refer="animation" instance="instSame">
       <property name="bounds"/>
     </media>
     <media id="icon" src="../media/icon.png" descriptor="iconDesc"/>
     <media id="shoes" src="../media/shoes.mp4" descriptor="shoesDesc"/>
     <link id="lIcon" xconnector="conEx#onBeginStart">
       <bind role="onBegin" component="reusedAnimation" interface="segIcon"/>
       <bind role="start" component="icon"/>
     </link>
     <link id="lBeginShoes" xconnector="conEx#onKeySelectionStopSetStart">
       <bind role="onSelection" component="icon">
         <bindParam name="keyCode" value="RED"/>
       </bind>
       <bind role="start" component="shoes"/>
       <bind role="set" component="reusedAnimation" interface="bounds">
         <bindParam name="varSet" value="5%,6.67%,45%,45%"/>
       </bind>
       <bind role="stop" component="icon"/>
       </link>
       <link id="lEndShoes" xconnector="conEx#onEndSet">
       <bind role="onEnd" component="shoes"/>
       <bind role="set" component="reusedAnimation" interface="bounds">
         <bindParam name="varSet" value="0,0,222.22%,222.22%"/>
       </bind>
     </link>
    </context>
    <link id="lMusic" xconnector="conEx#onBeginStartDelay">
       <bind role="onBegin" component="animation"/>
       <bind role="start" component="background">
         <bindParam name="delay" value="5s"/>
       </bind>
       <bind role="start" component="choro">
         <bindParam name="delay" value="5s"/>
       </bind>
     </link>
     <link id="lDrible" xconnector="conEx#onBeginStart">
       <bind role="onBegin" component="animation" interface="segDrible"/>
```

```
          <bind role="start" component="drible"/>
        </link>
        <link id="lPhoto" xconnector="conEx#onBeginStart">
          <bind role="onBegin" component="animation" interface="segPhoto"/>
          <bind role="start" component="photo"/>
        </link>
        <link id="lEnd" xconnector="conEx#onEndStop">
          <bind role="onEnd" component="animation"/>
          <bind role="stop" component="choro"/>
          <bind role="stop" component="background"/>
        </link>
  </body>
</ncl>
```

Listado 3.24 El primer *João* con reúso.

3.7 El canal de interactividad

Vamos ahora añadir en nuestro ejemplo el uso del canal de interactividad. En esta nueva versión del programa NCL, cuando el ícono del zapato sea seleccionado, no solo vamos a presentar la propaganda del zapato, sino también un formulario HTML. El formulario debidamente lleno, podrá ser enviado a la tienda de la propaganda por medio del canal de interactividad y traerá como respuesta la confirmación de la compra. La tienda habiendo recibido el formulario, puede después proporcionar la entrega del material adquirido.

La Figura 3.9 ilustra la visión estructural de la nueva versión de la aplicación. Note que las únicas modificaciones fueron: la introducción de un objeto multimedia que representa el formulario en el contexto de la propaganda; el incremento de un actor más en las relaciones que es disparado por la selección del ícono de la zapato, el cual permitirá exhibir el formulario; y el hecho de que ahora no es el final de la exhibición de la propaganda del zapato la que vuelve el video de la animación a su tamaño original, sino el final del llenado del formulario o el final del tiempo máximo para su llenado.

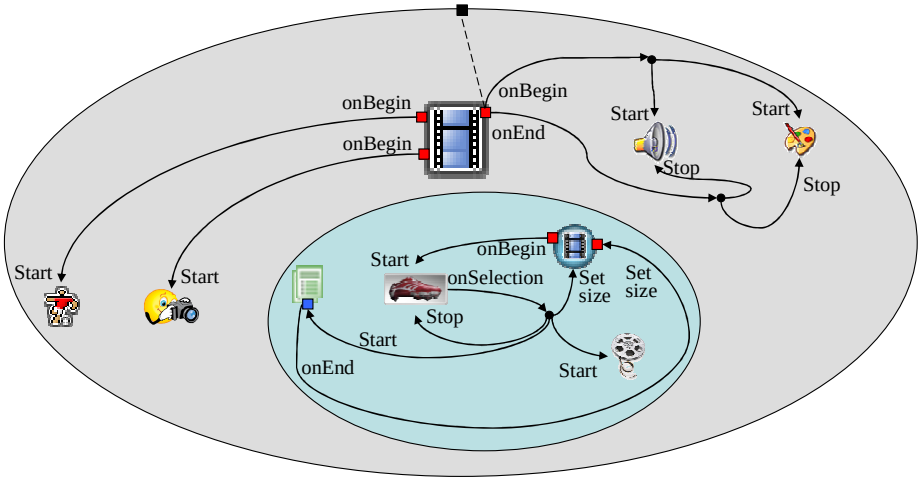


Figura 3.9 Visión estructural de la versión con el uso do canal de interactividad.

Debido a la introducción de un nuevo elemento <media>, tenemos también que introducir el elemento <region> especificando donde será presentado, y el elemento <descriptor>, creando la unión del elemento <media> con el elemento <region>. El Listado 3.25, ilustra los nuevos elementos insertados. Note que una vez más, la región definida para la exhibición del formulario es hija de la región definida para la exhibición del video de la animación, es decir, el posicionamiento de la primera es relativo a la segunda. Note también que en el descriptor fue especificado un tiempo máximo para llenar del formulario, que para este caso es de 45 segundos.

```
<head>
  <regionBase>
  ...
   <region id="screenReg" width="100%" height="100%" zIndex="2">
     ...
     <region id="formReg" left="56.25%" top="8.33%" width="38.75%"
                                        height="71.7%" zIndex="3"/>
     </region>
  </regionBase>
  <descriptorBase>
  ...
     <descriptor id="formDesc" region="formReg" focusIndex="1"
                                                explicitDur="45s"/>
  </descriptorBase>
  ...
</head>

<body>
  ...
  <context id="advert">
  ...
  <media id="ptForm" src="../media/ptForm.htm" descriptor="formDesc"/>
  ...
  </context>
  ...
</body>
```

Listado 3.25 Nuevos elementos para la presentación del formulario.

Aun en el Listado 3.25, note que en el elemento <descriptor> "formDesc" un nuevo atributo fue definido: *focusIndex*. Este atributo define el elemento que se encuentra en ese momento actico "en foco" para la navegación por las flechas del control remoto. En este, caso solo hay un elemento, pero cuando incrementemos más nuestro ejemplo, en la Sección 3.12, otros elementos en capacidad de obtener el foco serán definidos. Allí explicaremos mejor ese atributo. Un elemento en foco, cuando es presionada a tecla "ENTER", pasa a recibir la capacidad de navegación por el control remoto hasta que la tecla "BACK" sea presionada retornando entonces el control al intérprete NCL. En el caso en cuestión, después de ser presionada la tecla "ENTER", el formulario HTML puede ser llenado.

Debemos ahora alterar algunas relaciones de la versión anterior para obtener una nueva versión. La relación "lBeginShoes" debe incrementar un rol de acción para dar inicio a la presentación del formulario. La relación "lEndShoes" debe ser substituida por la relación "lEndForm", dado que ahora, es la finalización de la presentación del formulario la que devuelve el video de la animación para sus valores de posicionamiento iníciales, y no la finalización del video de propaganda del zapato. Las relaciones modificadas son presentadas en el Listado 3.26.

```
<link id="lBeginShoes" xconnector="conEx#onKeySelectionStopSetStart">
  <bind role="onSelection" component="icon">
    <bindParam name="keyCode" value="RED"/>
  </bind>
  <bind role="start" component="shoes"/>
  <bind role="start" component="ptForm"/>
  <bind role="set" component="reusedAnimation" interface="bounds">
    <bindParam name="varSet" value="5%,6.67%,45%,45%"/>
  </bind>
  <bind role="stop" component="icon"/>
</link>
<link id="lEndForm" xconnector="conEx#onEndSet">
  <bind role="onEnd" component="ptForm"/>
  <bind role="set" component="reusedAnimation" interface="bounds">
    <bindParam name="varSet" value="0,0,222.22%,222.22%"/>
  </bind>
</link>
```

Listado 3.26 Relaciones modificadas para la nueva versión.

Las especificaciones completas de la nueva versión con el uso del canal de interactividad son ilustradas en el Listado 3.27.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!--Ejemplo de canal de interactividad-->
<ncl id="canal_interactividad" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
 <head>
   <regionBase>
     <region id="backgroundReg" width="100%" height="100%" zIndex="1"/>
       <region id="screenReg" width="100%" height="100%" zIndex="2">
       <region id="frameReg" left="5%" top="6.7%" width="18.5%" height="18.5%"
                                                              zIndex="3"/>
       <region id="iconReg" left="87.5%" top="11.7%" width="8.45%" height="6.7%"
```

```
                                                                    zIndex="3"/>
      <region id="shoesReg" left="15%" top="60%" width="25%" height="25%"
                                                                    zIndex="3"/>
      <region id="formReg" left="56.25%" top="8.33%" width="38.75%"
                                                 height="71.7%" zIndex="3"/>
   </region>
  </regionBase>
  <descriptorBase>
      <descriptor id="backgroundDesc" region="backgroundReg"/>
      <descriptor id="screenDesc" region="screenReg"/>
      <descriptor id="photoDesc" region="frameReg" explicitDur="5s"/>
      <descriptor id="audioDesc"/>
      <descriptor id="dribleDesc" region="frameReg"/>
      <descriptor id="iconDesc" region="iconReg" explicitDur="6s"/>
      <descriptor id="shoesDesc" region="shoesReg"/>
      <descriptor id="formDesc" region="formReg" focusIndex="1"
                                           explicitDur="15s"/>
   </descriptorBase>
   <connectorBase>
      <importBase documentURI="../causalConnBase.ncl" alias="conEx"/>
   </connectorBase>
</head>
<body>
   <port id="entry" component="animation"/>
   <media id="background" src="../media/background.png"
                                      descriptor="backgroundDesc"/>
   <media id="animation" src="../media/animGar.mp4" descriptor="screenDesc">
       <area id="segDrible" begin="12s"/>
       <area id="segPhoto" begin="41s"/>
       <area id="segIcon" begin="45s" end="51s"/>
   </media>
   <media id="choro" src="../media/choro.mp3" descriptor="audioDesc"/>
   <media id="drible" src="../media/drible.mp4" descriptor="dribleDesc"/>
   <media id="photo" src="../media/photo.png" descriptor="photoDesc"/>
   <context id="advert">
     <media id="reusedAnimation" refer="animation" instance="instSame">
       <property name="bounds"/>
     </media>
     <media id="icon" src="../media/icon.png" descriptor="iconDesc"/>
     <media id="shoes" src="../media/shoes.mp4" descriptor="shoesDesc"/>
     <media id="ptForm" src="../media/ptForm.htm" descriptor="formDesc"/>
     <link id="lIcon" xconnector="conEx#onBeginStart">
       <bind role="onBegin" component="reusedAnimation" interface="segIcon"/>
       <bind role="start" component="icon"/>
     </link>
     <link id="lBeginShoes" xconnector="conEx#onKeySelectionStopSetStart">
       <bind role="onSelection" component="icon">
         <bindParam name="keyCode" value="RED"/>
       </bind>
       <bind role="start" component="shoes"/>
         <bind role="start" component="ptForm"/>
       <bind role="set" component="reusedAnimation" interface="bounds">
         <bindParam name="varSet" value="5%,6.67%,45%,45%"/>
       </bind>
       <bind role="stop" component="icon"/>
     </link>
     <link id="lEndForm" xconnector="conEx#onEndSet">
       <bind role="onEnd" component="ptForm"/>
       <bind role="set" component="reusedAnimation" interface="bounds">
        <bindParam name="varSet" value="0,0,222.22%,222.22%"/>
       </bind>
     </link>
   </context>
   <link id="lMusic" xconnector="conEx#onBeginStartDelay">
      <bind role="onBegin" component="animation"/>
      <bind role="start" component="background">
        <bindParam name="delay" value="5s"/>
      </bind>
      <bind role="start" component="choro">
        <bindParam name="delay" value="5s"/>
      </bind>
   </link>
```

```
    <link id="lDrible" xconnector="conEx#onBeginStart">
       <bind role="onBegin" component="animation" interface="segDrible"/>
       <bind role="start" component="drible"/>
    </link>
    <link id="lPhoto" xconnector="conEx#onBeginStart">
       <bind role="onBegin" component="animation" interface="segPhoto"/>
       <bind role="start" component="photo"/>
    </link>
    <link id="lEnd" xconnector="conEx#onEndStop">
       <bind role="onEnd" component="animation"/>
       <bind role="stop" component="background"/>
       <bind role="stop" component="choro"/>
    </link>
 </body>
</ncl>
```

Listado 3.27 El primer *João* con uso del canal de interactividad.

La Figura 3.10 ilustra el momento de la presentación de la propaganda de interactividad del zapato obtenida durante la ejecución de la aplicación en la implementación de referencia del middleware Ginga-NCL.

Figura 3.10 Escenas de la aplicación del primer *João* con uso del canal de interactividad.

3.8 Uso de múltiples dispositivos de exhibición

Podemos ahora alterar la versión anterior de la aplicación para trabajar con múltiples dispositivos de exhibición.

En esta nueva versión, vamos a exhibir toda la propaganda interactiva, es decir, el video de la propaganda y el formulario de compra en otro dispositivo diferente del usado para la presentación de los otros objetos multimedia.

Cuando hacemos uso de más de un dispositivo de exhibición, para cada una región base (elemento <regionBase>) debemos especificar a qué dispositivo las regiones definidas se refieren. Esto se realiza en el atributo

device del elemento <regionBase>, como ilustra el Listado 3.28. Cuando este argumento no es declarado, asume un valor *default*.

```
<regionBase>
  <region id="screenReg" width="100%" height="100%" zIndex="1">
     <region id="frameReg" left="5%" top="6.7%" width="18.5%"
                                    height="18.5%" zIndex="2"/>
     <region id="iconReg" left="87.5%" top="11.7%" width="8.45%"
                                    height="6.7%" zIndex="2"/>
  </region>
</regionBase>

<regionBase device="systemScreen(1)">
  <region id="backgroundReg" width="100%" height="100%" zIndex="1">
    <region id="shoesReg" left="0" top="25%" width="50%" height="50%"
                                                    zIndex="2"/>
    <region id="formReg" left="50%" top="0" width="50%" height="100%"
                                                    zIndex="2"/>
  </region>
</regionBase>
```

Listado 3.28 Uso de múltiples dispositivos de exhibición.

En el Listado 3.28, el video de la animación, el vídeo del drible, la foto del marcador caído y el ícono del zapato aparecerán en la pantalla del dispositivo-base de la presentación, declarado por *default*. En el caso del Sistema Brasileño de TV Digital Terrestre, por *default* el dispositivo-base es la pantalla de la TV. Ya la figura de fondo, la propaganda del zapato y el formulario a llenar aparecerán en un segundo dispositivo de exhibición; por ejemplo, la pantalla de un celular usado para interactividad.

El programa NCL debe ahora ser modificado ya que como no habrá más superposición de informaciones, no es necesario redimensionar el vídeo de la animación. Queda por cuenta del lector hacer esas modificaciones, dado que no proseguiremos más con ese ejemplo hasta el Capítulo 15. La Figura 3.11 ilustra el momento de la presentación de la propaganda, obtenido durante la ejecución de la aplicación en la implementación de referencia del middleware Ginga-NCL.

Figura 3.11 Escenas de la aplicación el primer *João* con el uso de múltiples dispositivos.

3.9 Adaptación de contenido

Ahora vamos a volver a la versión de la Sección 3.7 y en ella incrementar la facilidad de adaptación de contenido. Por lo tanto, vamos a exhibir el formulario en diferentes idiomas dependiendo del perfil del telespectador. Si el usuario habla el idioma inglés, vamos a presentar el formulario en inglés; caso contrario, presentaremos el formulario en español.

La visión estructural para esa versión del programa NCL es ilustrada en la Figura 3.12.

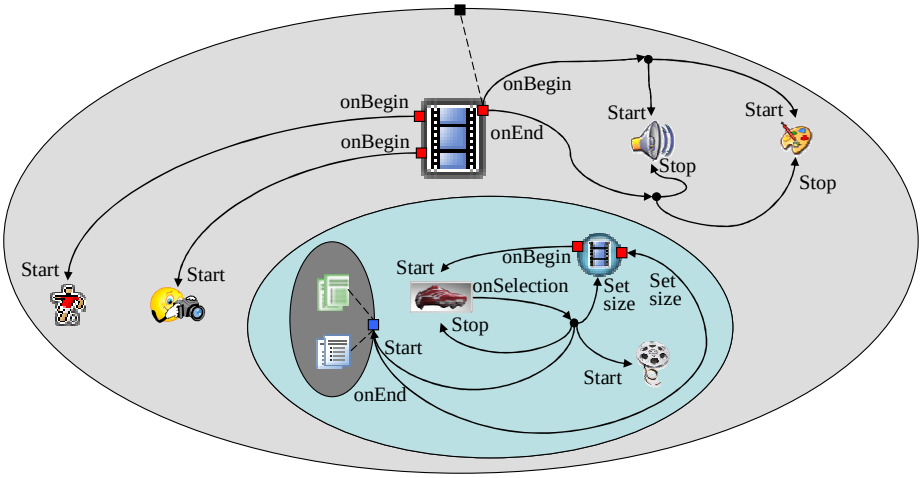


Figura 3.12 Visión estructural de la versión con adaptación de contenido.

Comparando la Figura 3.9 con la Figura 3.12, el lector puede observar que la única diferencia ocurre en el objeto multimedia que representa el formulario. En la Figura 3.12, este objeto es substituido por un elemento <switch>, representado en gris dentro del contexto de la propaganda. El elemento <switch>, a su vez, contiene tanto el objeto multimedia que representa el formulario en español como el objeto multimedia que representa el formulario en inglés. Todos las relaciones que tenían como actor al objeto multimedia formulario ("ptForm"), en la versión de la Sección 3.7, deben ahora referenciarse al elemento <switch> en la nueva versión del programa NCL.

El elemento <switch> escoge uno de sus componentes hijos para presentarlo, basado en un conjunto de reglas. Las reglas deben ser especificadas como hijas del elemento <ruleBase>, que a su vez es hijo del elemento <head>, conforme es ilustrado para el ejemplo en cuestión en el Listado 3.29.

```
<head>
   <ruleBase>
     <rule id="en" var="system.language" value="en" comparator="eq"/>
       </ruleBase>
        ...
   </head>
```

Listado 3.29 Elementos <ruleBase> y <rule>.

Un elemento <rule> tiene: un identificador, un atributo especificando una variable a ser probada (var), un atributo especificando el valor sobre el cual la variable debe ser probada (value) y un atributo especificando el operador de comparación. En el caso del Listado 3.29, la regla "en" es especificada para probar si la variable "system.language" tiene su valor igual a "en". La variable "system.language" es una variable controlada por el sistema receptor, que atribuye un valor a ella dependiendo del idioma del usuario telespectador. Son diversas las variables controladas por el sistema. En el Capítulo 9 discute el uso y el mantenimiento de esas variables.

La adaptación del contenido es especificada por el elemento <switch>, conforme se ilustra en el Listado 3.30 para la nueva versión del primer *João.*

```
<switch id="form">
  <switchPort id="spForm">
        <mapping component="enForm"/>
        <mapping component="ptForm"/>
  </switchPort>
  <bindRule constituent="enForm" rule="en"/>
  <defaultComponent component="ptForm"/>
  <media id="ptForm" src="../media/ptForm.htm" descriptor="formDesc"/>
  <media id="enForm" src="../media/enForm.htm" descriptor="formDesc"/>
</switch>
```

Listado 3.30 Elementos <switch> y sus elementos hijos.

Una relación no puede referenciar directamente a un elemento hijo del <switch>, pero puede hacerlo indirectamente por medio de puertos del switch. Un elemento <switchPort> define un puerto que puede ser usado para referenciar cualquier interface de un elemento hijo del <switch>, para lo que con un mapeo (elemento <mapping>) con el puerto fue establecido. En otras palabras, cuando la relación es establecida con un <switchPort>, la interface del elemento seleccionado por el *switch* y que haya mapeado con el puerto es la que participa como actor de la relación.

El elemento <bindRule> determina cual elemento hijo del <switch> es escogido, dependiendo de la regla definida por el elemento <rule>, ya discutida. Por ejemplo, en el Listado 3.30, si la regla "en" fuera atendida (recuerde que esa regla especificaba que el idioma del usuario telespectador era el inglés), el elemento <media> "enForm" deberá ser escogido.

Las reglas son evaluadas en secuencia, y la primera regla que atiende las exigencias tiene su componente seleccionado. En el caso que ninguna regla sea satisfecha, el componente definido en el elemento <defaultComponent> debe ser escogido. Un elemento <switch> puede tener como elementos hijos seleccionables a elementos <media> o <context>.

Como mencionamos anteriormente, esa nueva versión del programa NCL debe modificar sus relaciones para referenciar el elemento <switch> el cual fue definido en el Listado 3.30 y ya no directamente al elemento <media> "ptForm", conforme se ilustra en el Listado 3.31. Note la referencia al elemento <switch> "form" y a su puerto (<switchPort>) "spForm".

```
<link id="lBeginShoes" xconnector="conEx#onKeySelectionStopSetStart">
  <bind role="onSelection" component="icon">
        <bindParam name="keyCode" value="RED"/>
  </bind>
  <bind role="start" component="shoes"/>
  <bind role="start" component="form" interface="spForm"/>
  <bind role="set" component="reusedAnimation" interface="bounds">
        <bindParam name="varSet" value="5%,6.67%,45%,45%"/>
  </bind>
  <bind role="stop" component="icon"/>
</link>

<link id="lEndForm" xconnector="conEx#onEndSet">
  <bind role="onEnd" component="form" interface="spForm"/>
  <bind role="set" component="reusedAnimation" interface="bounds">
        <bindParam name="varSet" value="0,0,222.22%,222.22%"/>
  </bind>
</link>
```

Listado 3.31 Redefinición de las relaciones para referenciar al elemento <switch>.

La especificación completa de la nueva versión del programa NCL es ilustrada en el Listado 3.32.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Ejemplo de switch y reglas -->
```

```
<ncl id="switch" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
  <head>
    <ruleBase>
      <rule id="en" var="system.language" value="en" comparator="eq"/>
      <rule id="int" var="service.interactivity" value="true" comparator="eq"/>
    </ruleBase>
    <regionBase>
      <region id="backgroundReg" width="100%" height="100%" zIndex="1"/>
      <region id="screenReg" width="100%" height="100%" zIndex="2">
        <region id="frameReg" left="5%" top="6.7%" width="18.5%" height="18.5%"
                                                                  zIndex="3"/>
        <region id="iconReg" left="87.5%" top="11.7%" width="8.45%"
                                                       height="6.7%" zIndex="3"/>
        <region id="shoesReg" left="15%" top="60%" width="25%" height="25%"
                                                                  zIndex="3"/>
        <region id="formReg" left="56.25%" top="8.33%" width="38.75%"
                                                      height="71.7%" zIndex="3"/>
      </region>
    </regionBase>
    <descriptorBase>
       <descriptor id="backgroundDesc" region="backgroundReg"/>
       <descriptor id="screenDesc" region="screenReg"/>
       <descriptor id="photoDesc" region="frameReg" explicitDur="5s"/>
       <descriptor id="audioDesc"/>
       <descriptor id="dribleDesc" region="frameReg"/>
       <descriptor id="iconDesc" region="iconReg" explicitDur="6s"/>
       <descriptor id="shoesDesc" region="shoesReg"/>
       <descriptor id="formDesc" region="formReg" focusIndex="1"
                                                 explicitDur="15s"/>
    </descriptorBase>
    <connectorBase>
       <importBase documentURI="../causalConnBase.ncl" alias="conEx"/>
    </connectorBase>
 </head>
 <body>
       <port id="entry" component="animation"/>
       <media id="background" src="../media/background.png"
                                        descriptor="backgroundDesc"/>
       <media id="animation" src="../media/animGar.mp4" descriptor="screenDesc">
          <area id="segDrible" begin="12s"/> <area id="segPhoto" begin="41s"/>
          <area id="segIcon" begin="45s" end="51s"/>
       </media>
       <media id="choro" src="../media/choro.mp3" descriptor="audioDesc"/>
       <media id="drible" src="../media/drible.mp4" descriptor="dribleDesc"/>
       <media id="photo" src="../media/photo.png" descriptor="photoDesc"/>
       <context id="advert">
         <media id="reusedAnimation" refer="animation" instance="instSame">
           <property name="bounds"/>
         </media>
         <media id="icon" src="../media/icon.png" descriptor="iconDesc"/>
         <media id="shoes" src="../media/shoes.mp4" descriptor="shoesDesc"/>
         <switch id="form">
            <switchPort id="spForm">
              <mapping component="enForm"/>
              <mapping component="ptForm"/>
            </switchPort>
            <bindRule constituent="enForm" rule="en"/>
            <defaultComponent component="ptForm"/>
            <media id="ptForm" src="../media/ptForm.htm" descriptor="formDesc"/>
            <media id="enForm" src="../media/enForm.htm" descriptor="formDesc"/>
         </switch>
         <link id="lIcon" xconnector="conEx#onBeginStart">
            <bind role="onBegin" component="reusedAnimation"interface="segIcon"/>
            <bind role="start" component="icon"/>
         </link>
         <link id="lBeginShoes" xconnector="conEx#onKeySelectionStopSetStart">
             <bind role="onSelection" component="icon">
                 <bindParam name="keyCode" value="RED"/>
             </bind>
             <bind role="start" component="shoes"/>
             <bind role="start" component="form" interface="spForm"/>
             <bind role="set" component="reusedAnimation" interface="bounds">
```

```
                <bindParam name="varSet" value="5%,6.67%,45%,45%"/>
            </bind>
            <bind role="stop" component="icon"/>
        </link>
        <link id="lEndForm" xconnector="conEx#onEndSet">
            <bind role="onEnd" component="form" interface="spForm"/>
            <bind role="set" component="reusedAnimation" interface="bounds">
                 <bindParam name="varSet" value="0,0,222.22%,222.22%"/>
            </bind>
        </link>
    </context>
    <link id="lMusic" xconnector="conEx#onBeginStartDelay">
        <bind role="onBegin" component="animation"/>
        <bind role="start" component="background">
           <bindParam name="delay" value="5s"/>
        </bind>
        <bind role="start" component="choro">
           <bindParam name="delay" value="5s"/>
        </bind>
    </link>
    <link id="lDrible" xconnector="conEx#onBeginStart">
        <bind role="onBegin" component="animation" interface="segDrible"/>
        <bind role="start" component="drible"/>
    </link>
    <link id="lPhoto" xconnector="conEx#onBeginStart">
        <bind role="onBegin" component="animation" interface="segPhoto"/>
        <bind role="start" component="photo"/>
    </link>
    <link id="lEnd" xconnector="conEx#onEndStop">
        <bind role="onEnd" component="animation"/>
        <bind role="stop" component="background"/>
        <bind role="stop" component="choro"/>
    </link>
  </body>
</ncl>
```

Listado 3.32 El primer *João* con adaptación de contenido.

3.10 Uso del nodo settings

Ahora vamos a permitir en nuestra aplicación el primer *João* que todas las propagandas interactivas (y solo ellas) sean habilitadas o deshabilitadas por el telespectador. Si son deshabilitadas, por ejemplo, el ícono del zapato aparecerá, imposibilitando el accionamiento de la propaganda correspondiente.

Para hacer el control de la interactividad, vamos a definir una variable global del programa (servicio) denominada “service.interactivity”. Conforme el valor de esa variable, las propagandas interactivas serán deshabilitadas (service.interactivity = “false”) o habilitadas (service.interactivity = “true”). La verdad, el valor de la variable será usado (probado en las relaciones) para permitir o no el aparecimiento de los íconos que permiten la propaganda interactiva (en el caso de nuestra aplicación, el ícono del zapato).

En una aplicación NCL, las variables globales son tratadas como propiedades del nodo *settings*. En NCL, ese nodo es representado por el elemento <media>, con el atributo *type* igual a “application/x-ncl-settings”. Propiedades cuyos valores pueden ser manipulados por una aplicación NCL

deben ser declaradas en el elemento <property> hijo del elemento <media> representando el nodo settings (excepto cuando la manipulación fuera realizada por las reglas, como lo discutiremos en el Capítulo 11). Así, para nuestra nueva versión de la aplicación, el nodo settings debe ser declarado y su valor iniciado, como se ilustra en el Listado 3.33.

```
<media id="globalVar" type="application/x-ncl-settings">
  <property name="service.interactivity" value="true"/>
</media>
```

Listado 3.33 El elemento <media> del tipo “application/x-ncl-settings”.

Para el control de las propagandas interactivas, vamos a definir un contexto de interactividad, así pues, será posible su reutilización en otras aplicaciones, lo que tornará nuestro programa mejor estructurado. El elemento <context> “interactivity” contendrá: el nodo settings, un ícono para avisar al usuario telespectador que la interactividad está activa y otro para alertar que ella está inhibida. Conforme el usuario seleccione un ícono, él es substituido por otro, permitiendo así al usuario habilitar y deshabilitar la interactividad. Cada vez que el ícono es cambiado, la variable “service.interactivity” cambia de valor.

Como, en el inicio de la aplicación, el ícono que informa que la interactividad está habilitada (iniciación del procedimiento) debe ser exhibido y la variable colocada en “true”, vamos, para facilitar la estructuración, a colocar un elemento <media> en el contexto de interactividad representando la misma instancia de presentación de la película de la animación, que dará partida al procedimiento de iniciación. La colocación de ese elemento de hecho dificulta la reutilización del contexto en otra aplicación, pero vamos a usar el procedimiento de todas maneras para una vez más ejemplificar el mecanismo de reúso.

La Figura 3.13 ilustra la visión estructural del contexto de interactividad.

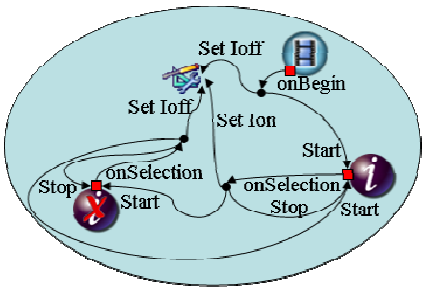


Figura 3.13 Visión estructural del contexto de interactividad.

El lector debe notar la existencia de tres relaciones. La primera inicia la variable “service.interactivity” y la exhibición del ícono “intOn”. La

segunda, cuando la selección del ícono “intOn” por la tecla INFO, para su exhibición, inicia la presentación del ícono “intOff” y cambia el valor de “service.interactivity”. La tercera es parecida con la segunda, cuando de la selección del ícono “intOff” por la tecla INFO, para su exhibición, inicia la presentación del ícono “intOn” y cambia el valor de “service.interactivity”. En el Listado 3.34 presenta el código NCL correspondiente.

```
<context id="interactivity">
   <media id="globalVar" type="application/x-ncl-settings">
      <property name="service.interactivity" value="true"/>
   </media>
   <media id="anotherAnimation" refer="animation"
                                              instance="instSame"/>
   <media id="intOn" src="../media/intOn.png"
                                              descriptor="intDesc"/>
   <media id="intOff" src="../media/intOff.png"
                                              descriptor="intDesc"/>
   <link id="lInt" xconnector="conEx#onBeginSetStart">
      <bind role="onBegin" component="anotherAnimation"/>
      <bind role="start" component="intOn"/>
      <bind role="set" component="globalVar"
                              interface="service.interactivity">
         <bindParam name="var" value="true"/>
      </bind>
   </link>
   <link id="lOn" xconnector="conEx#onKeySelectionStopSetStart">
      <bind role="onSelection" component="intOn">
         <bindParam name="keyCode" value="INFO"/>
      </bind>
      <bind role="start" component="intOff"/>
      <bind role="stop" component="intOn"/>
      <bind role="set" component="globalVar"
                              interface="service.interactivity">
         <bindParam name="varSet" value="false"/>
      </bind>
   </link>
   <link id="lOff" xconnector="conEx#onKeySelectionStopSetStart">
      <bind role="onSelection" component="intOff">
         <bindParam name="keyCode" value="INFO"/>
      </bind>
      <bind role="start" component="intOn"/>
      <bind role="stop" component="intOff"/>
      <bind role="set" component="globalVar"
                              interface="service.interactivity">
         <bindParam name="varSet" value="true"/>
      </bind>
   </link>
</context>
```

Listado 3.34 Contexto de interactividad.

Ahora que ya tenemos la manipulación de la variable “service.interactivity” definida, vamos a usar su valor para controlar que aparezca o no del ícono del zapato, que habilitará la propaganda de interactividad. Para lo cual, tenemos que modificar la relación que da inicio a la presentación del ícono, para que teste antes el

valor de la variable “service.interactivity”. Como la variable es una propiedad de los settings, vamos a reusarla (aquí, si un ejemplo de reutilización bien útil) dentro del contexto de la propaganda como el actor (de hecho sus propiedades “service.interactivity” de la relación que dispara la presentación del ícono. La Figura 3.14 ilustra la visión estructural de toda la nueva versión del primer *João*. Note que un nuevo elemento <bind> fue adicionado al enlace que finaliza todos los objetos al término del video de la animación. Este nuevo elemento <bind> es conectado al contexto de interactividad sin especificar ningún puerto de entrada. Eso significa en NCL, que la acción “stop” del enlace será aplicada a todos los objetos del contexto que aun estén siendo presentados.

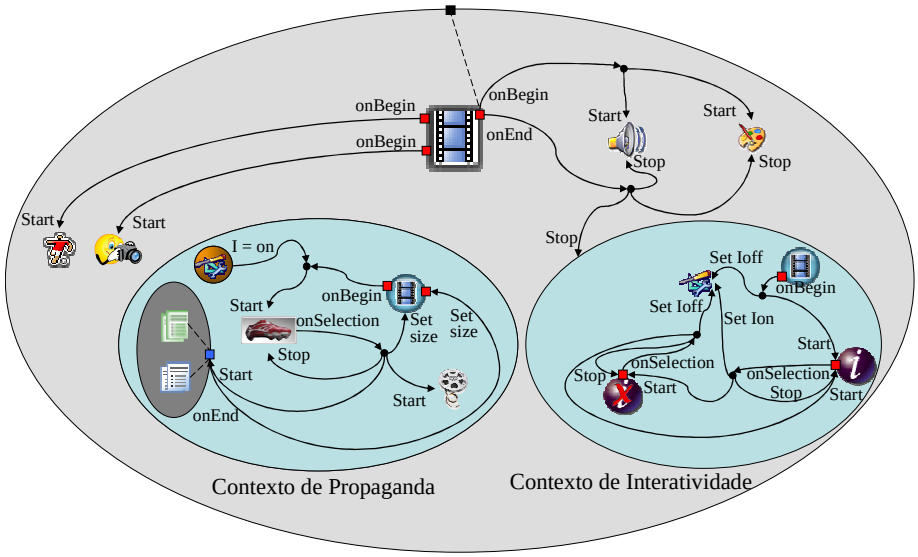


Figura 3.14 Visión estructural de la nueva versión del primer *João*.

El lector debe observar, por la comparación del contexto de propaganda de la Figura 3.12 con la de la Figura 3.14, que apenas una relación cambió. Todo lo demás permanece igual. La nueva relación debe testar el valor de la variable “service.interactivity” para iniciar o no la presentación del ícono del zapato, conforme se ilustra en el Listado 3.35.

```
<context id="advert">
...
    <media id="reusedGlobalVar" refer="globalVar"
                                              instance="instSame"/>
...
    <link id="lIcon" xconnector="conEx#onBeginVarStart">
       <bind role="onBegin" component="reusedAnimation"
                                              interface="segIcon"/>
       <bind role="var" component="reusedGlobalVar"
                              interface="service.interactivity"/>
```

```
        <bind role="start" component="icon"/>
    </link>
...
</context>
```

Listado 3.35 Nueva relación para la exhibición del ícono del zapato (“icon”).

Vale la pena abrir aquí un paréntesis para ver la definición del elemento <causalConnector> “onBeginVarStart”, especificado externamente, como ilustra el Listado 3.36.

```
<causalConnector id="onBeginVarStart">
   <compoundCondition operator="and">
      <simpleCondition role="onBegin"/>
      <assessmentStatement comparator="eq">
       <attributeAssessment role="var"
             attributeType="nodeProperty" eventType="attribution"/>
       <valueAssessment value="true"/>
      </assessmentStatement>
   </compoundCondition>
   <simpleAction role="start"/>
</causalConnector>
```

Listado 3.36 Elemento <connector> “onBeginVarStart”.

En el Listado 3.36 la condición de disparo de la relación es compuesta. El disparo ocurre si fuera iniciada la presentación de una interface correspondiente al rol “onBegin” (en la relación del Listado 3.30, cuando fuera presentado en el instante del video de la animación especificado para que aparezca el ícono) y si la variable correspondiente al rol “var” (en la relación del Listado 3.30 de la variable “service.interactivity”) tuviera el valor atribuido igual a “true”. Valores de las variables son testados a partir de los elementos <assessmentStatement>, conforme veremos en el Capítulo 10. Cuando la condición compuesta fuera satisfecha, será disparada la acción que inicia la presentación del rol “start” (en la relación del Listado 3.30, asociada al ícono “icon”).

Finalmente, note que para la presentación del ícono, deben ser definidos el elemento <region> para su posicionamiento y el elemento <descriptor> conectando el elemento <region> al elemento <media> que él representa. El Listado 3.37 presenta el código NCL correspondiente a esta nueva versión del primer *João.*

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Ejemplo de uso y reúso de settings y link probando variable settings -->
<ncl id="settings" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
 <head>
   <ruleBase>
     <rule id="en" var="system.language" value="en" comparator="eq"/>
   </ruleBase>
   <regionBase>
     <region id="backgroundReg" width="100%" height="100%" zIndex="1"/>
     <region id="screenReg" width="100%" height="100%" zIndex="2">
       <region id="frameReg" left="5%" top="6.7%" width="18.5%" height="18.5%"
                                                                  zIndex="3"/>
       <region id="iconReg" left="87.5%" top="11.7%" width="8.45%" height="6.7%"
```

```
                                                                    zIndex="3"/>
      <region id="shoesReg" left="15%" top="60%" width="25%" height="25%"
                                                                    zIndex="3"/>
      <region id="formReg" left="56.25%" top="8.33%" width="38.75%"
                                                        height="71.7%" zIndex="3"/>
      <region id="intReg" left="92.5%" top="91.7%" width="5.07%" height="6.51%"
                                                                    zIndex="3"/>
    </region>
  </regionBase>
  <descriptorBase>
    <descriptor id="backgroundDesc" region="backgroundReg"/>
    <descriptor id="screenDesc" region="screenReg"/>
    <descriptor id="photoDesc" region="frameReg" explicitDur="5s"/>
    <descriptor id="audioDesc"/>
    <descriptor id="dribleDesc" region="frameReg"/>
    <descriptor id="iconDesc" region="iconReg" explicitDur="6s"/>
    <descriptor id="shoesDesc" region="shoesReg"/>
    <descriptor id="formDesc" region="formReg" focusIndex="1"
                                          explicitDur="15s"/>
    <descriptor id="intDesc" region="intReg"/>
  </descriptorBase>
  <connectorBase>
    <importBase documentURI="../causalConnBase.ncl" alias="conEx"/>
  </connectorBase>
</head>

<body>
  <port id="entry" component="animation"/>
  <media id="background" src="../media/background.png"
                                        descriptor="backgroundDesc"/>
  <media id="animation" src="../media/animGar.mp4"
                                        descriptor="screenDesc">
       <area id="segDrible" begin="12s"/>
       <area id="segPhoto" begin="41s"/>
       <area id="segIcon" begin="45s" end="51s"/>
  </media>
  <media id="choro" src="../media/choro.mp3" descriptor="audioDesc"/>
    <media id="drible" src="../media/drible.mp4" descriptor="dribleDesc"/>
    <media id="photo" src="../media/photo.png" descriptor="photoDesc"/>
    <context id="interactivity">
     <media id="globalVar" type="application/x-ncl-settings">
       <property name="service.interactivity" value="true"/>
     </media>
     <media id="anotherAnimation" refer="animation" instance="instSame"/>
     <media id="intOn" src="../media/intOn.png" descriptor="intDesc"/>
     <media id="intOff" src="../media/intOff.png" descriptor="intDesc"/>
     <link id="lInt" xconnector="conEx#onBeginSetStart">
      <bind role="onBegin" component="anotherAnimation"/>
      <bind role="start" component="intOn"/>
      <bind role="set" component="globalVar" interface="service.interactivity">
        <bindParam name="var" value="true"/>
      </bind>
     </link>
     <link id="lOn" xconnector="conEx#onKeySelectionStopSetStart">
      <bind role="onSelection" component="intOn">
       <bindParam name="keyCode" value="INFO"/>
      </bind>
      <bind role="start" component="intOff"/>
      <bind role="stop" component="intOn"/>
      <bind role="set" component="globalVar" interface="service.interactivity">
       <bindParam name="varSet" value="false"/>
      </bind>
     </link>
     <link id="lOff" xconnector="conEx#onKeySelectionStopSetStart">
    <bind role="onSelection" component="intOff">
      <bindParam name="keyCode" value="INFO"/>
    </bind>
    <bind role="start" component="intOn"/>
      <bind role="stop" component="intOff"/>
    <bind role="set" component="globalVar" interface="service.interactivity">
      <bindParam name="varSet" value="true"/>
    </bind>
```

```
    </link>
    </context>
      <context id="advert">
      <media id="reusedAnimation" refer="animation" instance="instSame">
        <property name="bounds"/>
      </media>
      <media id="reusedGlobalVar" refer="globalVar" instance="instSame"/>
      <media id="icon" src="../media/icon.png" descriptor="iconDesc"/>
      <media id="shoes" src="../media/shoes.mp4" descriptor="shoesDesc"/>
        <switch id="form">
         <switchPort id="spForm">
            <mapping component="enForm"/>
            <mapping component="ptForm"/>
         </switchPort>
         <bindRule constituent="enForm" rule="en"/>
         <defaultComponent component="ptForm"/>
         <media id="ptForm" src="../media/ptForm.htm" descriptor="formDesc"/>
         <media id="enForm" src="../media/enForm.htm" descriptor="formDesc"/>
        </switch>
        <link id="lIcon" xconnector="conEx#onBeginVarStart">
          <bind role="onBegin" component="reusedAnimation" interface="segIcon"/>
          <bind role="var" component="reusedGlobalVar"
                                              interface="service.interactivity"/>
          <bind role="start" component="icon"/>
        </link>
        <link id="lBeginShoes" xconnector="conEx#onKeySelectionStopSetStart">
          <bind role="onSelection" component="icon">
            <bindParam name="keyCode" value="RED"/>
          </bind>
          <bind role="start" component="shoes"/>
          <bind role="start" component="form" interface="spForm"/>
          <bind role="set" component="reusedAnimation" interface="bounds">
            <bindParam name="varSet" value="5%,6.67%,45%,45%"/>
          </bind>
          <bind role="stop" component="icon"/>
         </link>
         <link id="lEndForm" xconnector="conEx#onEndSet">
            <bind role="onEnd" component="form" interface="spForm"/>
            <bind role="set" component="reusedAnimation" interface="bounds">
              <bindParam name="varSet" value="0,0,222.22%,222.22%"/>
            </bind>
         </link>
      </context>
      <link id="lMusic" xconnector="conEx#onBeginStartDelay">
         <bind role="onBegin" component="animation"/>
         <bind role="start" component="background">
            <bindParam name="delay" value="5s"/>
         </bind>
         <bind role="start" component="choro">
            <bindParam name="delay" value="5s"/>
         </bind>
      </link>
      <link id="lDrible" xconnector="conEx#onBeginStart">
         <bind role="onBegin" component="animation" interface="segDrible"/>
         <bind role="start" component="drible"/>
      </link>
      <link id="lPhoto" xconnector="conEx#onBeginStart">
         <bind role="onBegin" component="animation" interface="segPhoto"/>
         <bind role="start" component="photo"/>
      </link>
      <link  id="lEnd" xconnector="conEx#onEndStop">
         <bind role="onEnd" component="animation"/>
         <bind role="stop" component="background"/>
         <bind role="stop" component="choro"/>
         <bind role="stop" component="interactivity"/>
      </link>
    </body>
</ncl>
```

Listado 3.37 El primer *João* con el control de propagandas interactivas.

La Figura 3.15 ilustra el momento de la presentación del ícono de interactividad y en otro momento en que la interactividad fue inhibida, obtenida durante la ejecución de la aplicación en la implementación de la referencia del middleware Ginga-NCL.

Figura 3.15 Escenas de la aplicación del primer *João* con control de interactividad.

3.11 Efectos de transición y animación

Ahora introduciremos un efecto de transición cuando aparece el drible de vaivén de Garrincha (elemento <media> “drible”). El efecto de transición es una característica de presentación y por tanto, definido en el elemento <descriptor> asociado a el elemento <media> donde se quiere la transición.

Antes de asociar transiciones al elemento <descriptor> debemos crear una base de transiciones para aplicaciones NCL en cuestión. Eso es hecho a través del elemento <transitionBase>, también hijo del elemento <head>. Una base de transiciones contienen como dice su propio nombre, transiciones especificadas por el elemento <transition>. El Listado 3.38 ilustra dos tipos de transiciones que usaremos en nuestra aplicación. La primera transición identificada como “trans1”, especifica un desvanecimiento de 2 segundos. La segunda transición identificada como “trans2”, especifica un efecto barredera de barra vertical durante 1 segundo.

```
<head>
...
        <transitionBase>
                <transition id="trans1" type="fade" dur="2s"/>
                <transition id="trans2" type="barWipe" dur="1s"/>
        </transitionBase>
...
</head>
```

Listado 3.38 Elementos <transitionBase> y <transition>.

Ahora podemos asociar las transiciones a el elemento <descriptor> asociado a el video “drible” conforme ilustra el Listado 3.39

```
<descriptor id="dribleDesc" region="frameReg" transIn="trans1"
                                              transOut="trans2"/>
```

Listado 3.39 Efecto de transición.

En el Listado 3.39, una transición de desvanecimiento de la entrada del video “drible y un barrido vertical al final del video fueron definidos.

En esta misma versión de la aplicación, vamos incluir un efecto de animación a la foto del marcador caído en el suelo (elemento <media> “photo”) y también incluiremos un efecto de transparencia en esta foto. El Listado 3.40 ilustra como el descriptor asociado a la foto puede ser alterado para producir el efecto de transparencia (una transparencia de 60% fue definida).

```
<descriptor id="photoDesc" region="frameReg" explicitDur="5s">
       <descriptorParam name="transparency" value="0.6"/>
</descriptor>
```

Listado 3.40 Efecto de transparencia.

El efecto de animación que queremos es bien simple. Vamos hacer que la foto sea presentada y después se desplace vertical y continuamente. Por tanto tenemos que modificar la relación que define el inicio de la presentación de la foto. Vamos incrementando la relación en otra acción, una acción de atribución de valores, que modificará el valor del atributo *top* del elemento continuamente, hasta un cierto valor final. La Figura 3.16 ilustra la nueva visión estructural, donde, por claridad, los contextos de interactividad y de propaganda son presentados de forma resumida.

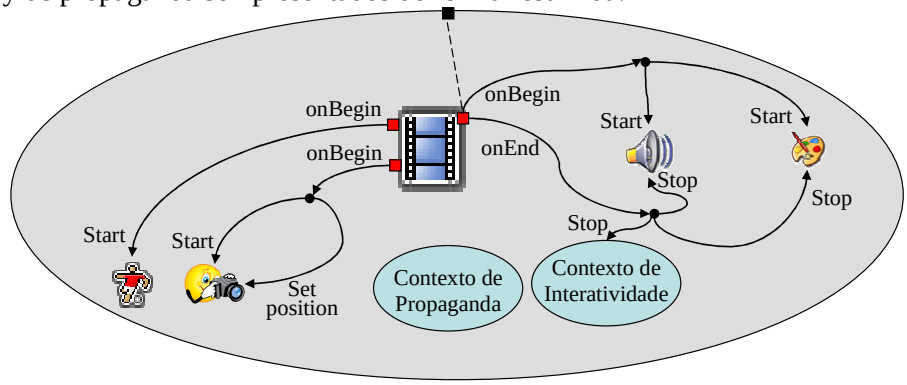


Figura 3.16 Visión estructural del primer *João*, con efecto de animación e transición.

Como modificaremos el atributo *top*, él necesita ser explícitamente declarado en el elemento <media> “photo”, conforme se ilustra en el Listado 3.41, que también presenta la definición de la nueva relación.

```
<body>
...
    <media id="photo" src="../media/photo.png"
                                        descriptor="photoDesc">
       <property name="top"/>
    </media>
...
    <link id="lPhoto" xconnector="conEx#onBeginStartSetDelay">
       <bind role="onBegin" component="animation"
                                           interface="segPhoto"/>
       <bind role="start" component="photo"/>
       <bind role="set" component="photo" interface="top">
       <bindParam name="var" value="290"/>
          <bindParam name="delay" value="1s"/>
          <bindParam name="duration" value="3s"/>
       </bind>
    </link>
...
<body>
```

Listado 3.41 Nueva relación con animación.

En el Listado 3.41, se puede observar el nuevo rol introducido (rol “set”). El establece que después de transcurrido 1 segundo del inicio de la exhibición de la foto (parámetro “delay”), o su valor del tope de cambiar para 290 pixeles continuamente, en un intervalo de 3 segundo (parámetro “duration”). El efecto que se tendrá es el de la foto se desplaza verticalmente.

La especificación completa de la nueva versión del programa NCL es ilustrada en el Listado 3.42.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Ejemplo de animación -->
<ncl id="animacion" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
 <head>
   <ruleBase>
     <rule id="en" var="system.language" value="en" comparator="eq"/>
     <rule id="int" var="service.interactivity" value="true" comparator="eq"/>
   </ruleBase>
   <transitionBase>
     <transition id="trans1" type="fade" dur="2s"/>
     <transition id="trans2" type="barWipe" dur="1s"/>
   </transitionBase>
   <regionBase>
     <region id="backgroundReg" width="100%" height="100%" zIndex="1"/>
     <region id="screenReg" width="100%" height="100%" zIndex="2">
       <region id="frameReg" left="5%" top="6.7%" width="18.5%" height="18.5%"
                                                          zIndex="3"/>
       <region id="iconReg" left="87.5%" top="11.7%" width="8.45%" height="6.7%"
                                                          zIndex="3"/>
       <region id="shoesReg" left="15%" top="60%" width="25%" height="25%"
                                                          zIndex="3"/>
       <region id="formReg" left="57.25%" top="9.83%" width="37.75%"
                                                 height="70.2%" zIndex="3"/>
       <region id="intReg" left="92.5%" top="91.7%" width="5.07%" height="6.51%"
                                                          zIndex="3"/>
```

```
      </region>
      </regionBase>
      <descriptorBase>
        <descriptor id="backgroundDesc" region="backgroundReg"/>
        <descriptor id="screenDesc" region="screenReg"/>
        <descriptor id="photoDesc" region="frameReg" explicitDur="5s">
          <descriptorParam name="transparency" value="0.6"/>
        </descriptor>
        <descriptor id="audioDesc"/>
        <descriptor id="dribleDesc" region="frameReg" transIn="trans1"
                                                          transOut="trans2"/>

        <descriptor id="iconDesc" region="iconReg" explicitDur="6s"/>
        <descriptor id="shoesDesc" region="shoesReg"/>
        <descriptor id="formDesc" region="formReg" focusIndex="1"
                                                          explicitDur="15s"/>
        <descriptor id="intDesc" region="intReg"/>
      </descriptorBase>
      <connectorBase>
        <importBase documentURI="../causalConnBase.ncl" alias="conEx"/>
      </connectorBase>
    </head>

    <body>
      <port id="entry" component="animation"/>
      <media id="background" src="../media/background.png"
                                                 descriptor="backgroundDesc"/>
      <media id="animation" src="../media/animGar.mp4" descriptor="screenDesc">
        <area id="segDrible" begin="11.5s"/>
        <area id="segPhoto" begin="41s"/>
        <area id="segIcon" begin="45s" end="51s"/>
      </media>
      <media id="choro" src="../media/choro.mp3" descriptor="audioDesc"/>
      <media id="drible" src="../media/drible.mp4" descriptor="dribleDesc"/>
      <media id="photo" src="../media/photo.png" descriptor="photoDesc">
        <property name="top"/>
      </media>
      <context id="interactivity">
        <media id="globalVar" type="application/x-ginga-settings">
          <property name="service.interactivity" value="true"/>
        </media>
        <media id="anotherAnimation" refer="animation" instance="instSame"/>
        <media id="intOn" src="../media/intOn.png" descriptor="intDesc"/>
        <media id="intOff" src="../media/intOff.png" descriptor="intDesc"/>
        <link id="lInt" xconnector="conEx#onBeginSetStart">
          <bind role="onBegin" component="anotherAnimation"/>
          <bind role="start" component="intOn"/>
          <bind role="set" component="globalVar"
                                          interface="service.interactivity">
          <bindParam name="var" value="true"/>
        </bind>
       </link>
       <link id="lOn" xconnector="conEx#onKeySelectionStopSetStart">
        <bind role="onSelection" component="intOn">
          <bindParam name="keyCode" value="INFO"/>
        </bind>
        <bind role="start" component="intOff"/>
        <bind role="stop" component="intOn"/>
        <bind role="set" component="globalVar"
                                    interface="service.interactivity">
          <bindParam name="varSet" value="false"/>
        </bind>
       </link>
       <link id="lOff" xconnector="conEx#onKeySelectionStopSetStart">
        <bind role="onSelection" component="intOff">
          <bindParam name="keyCode" value="INFO"/>
        </bind>
        <bind role="start" component="intOn"/>
        <bind role="stop" component="intOff"/>
        <bind role="set" component="globalVar"
```

```
                                interface="service.interactivity">
          <bindParam name="varSet" value="true"/>
        </bind>
      </link>
      </context>
        <context id="advert">
        <media id="reusedAnimation" refer="animation" instance="instSame">
         <property name="bounds"/>
        </media>
        <media id="reusedGlobalVar" refer="globalVar" instance="instSame"/>
        <media id="icon" src="../media/icon.png" descriptor="iconDesc"/>
        <media id="shoes" src="../media/shoes.mp4" descriptor="shoesDesc"/>
        <switch id="form">
          <switchPort id="spForm">
          <mapping component="enForm"/>
          <mapping component="ptForm"/>
          </switchPort>
            <bindRule constituent="enForm" rule="en"/>
            <defaultComponent component="ptForm"/>
            <media id="ptForm" src="../media/ptForm.htm" type="text/html"
                                                  descriptor="formDesc"/>
            <media id="enForm" src="../media/enForm.htm" type="text/html"
                                                  descriptor="formDesc"/>
          </switch>
          <link  id="lIcon" xconnector="conEx#onBeginVarStart">
            <bind role="onBegin" component="reusedAnimation"
                                                    interface="segIcon"/>
            <bind role="var" component="reusedGlobalVar"
                                       interface="service.interactivity"/>
            <bind role="start" component="icon"/>
        </link>
        <link  id="lBegingShoes" xconnector="conEx#onKeySelectionStopSetStart">
            <bind role="onSelection" component="icon">
                <bindParam name="keyCode" value="RED"/>
            </bind>
            <bind role="start" component="shoes"/>
            <bind role="start" component="form" interface="spForm"/>
            <bind role="set" component="reusedAnimation" interface="bounds">
                <bindParam name="varSet" value="5%,6.67%,45%,45%"/>
            </bind>
            <bind role="stop" component="icon"/>
        </link>
        <link  id="lEndForm" xconnector="conEx#onEndSet">
            <bind role="onEnd" component="form" interface="spForm"/>
            <bind role="set" component="reusedAnimation" interface="bounds">
                <bindParam name="varSet" value="0,0,222.22%,222.22%"/>
            </bind>
        </link>
        </context>
         <link  id="lMusic" xconnector="conEx#onBeginStartDelay">
           <bind role="onBegin" component="animation"/>
           <bind role="start" component="background">
             <bindParam name="delay" value="5s"/>
           </bind>
           <bind role="start" component="choro">
             <bindParam name="delay" value="5s"/>
           </bind>
        </link>
        <link  id="lDrible" xconnector="conEx#onBeginStart">
           <bind role="onBegin" component="animation" interface="segDrible"/>
           <bind role="start" component="drible"/>
        </link>
        <link  id="lPhoto" xconnector="conEx#onBeginStartSetDelay">
           <bind role="onBegin" component="animation" interface="segPhoto"/>
           <bind role="start" component="photo"/>
           <bind role="set" component="photo" interface="top">
              <bindParam name="var" value="290"/>
              <bindParam name="delay" value="1s"/>
              <bindParam name="duration" value="3s"/>
           </bind>
        </link>
        <link  id="lEnd" xconnector="conEx#onEndStop">
```

```
            <bind role="onEnd" component="animation"/>
            <bind role="stop" component="background"/>
            <bind role="stop" component="choro"/>
            <bind role="stop" component="interactivity"/>
         </link>
   </body>
</ncl>
```

Listado 3.42 El primer *João* con efectos de animación y transición.

La Figura 3. 17 ilustra el efecto de transición de salida, del tipo *bar wipe*, en el vídeo del esquina superior izquierda, que se sobrepone a el vídeo da animación.

Figura 3.17 Escenas de la aplicación el primer *João* con efectos de transición

3.12 Navegación por teclas

La próxima funcionalidad que agregaremos en nuestra aplicación del primer *João* es la opción de escoger la música de fondo, substituyendo el audio “choro” por otros llamados: “rock”, “techno” o “cartoon”.

El escoger los ritmos se creará un menú de navegación sobre íconos con las imágenes que los representan. La navegación se dará por medio de las teclas CURSOR_RIGHT y CURSOR_LEFT del control remoto. El ícono enfado, se selecciona por la tecla ENTER, lo que efectuará la substitución de la música.

Como los íconos estarán siempre visibles, para no sobreponerlos al video de la animación, vamos a redimensionar la región donde el video será presentado. Tenemos también de definir las regiones donde los íconos serán exhibidos. Así el nuevo elemento <regionBase> puede ser definida conforme se ilustra en el Listado 3.43. Debemos notar, que en la nueva definición, la totalidad de las regiones son definidas en relación a la región de la figura del fondo y que los íconos corresponden a los ritmos de la música de fondo serán presentados en la parte inferior de la pantalla.

```
<regionBase>
   <region id="backgroundReg" width="100%" height="100%"
                                                  zIndex="1">
      <region id="screenReg" width="100%" height="88%" zIndex="2"/>
```

```
      <region id="frameReg" left="5%" top="6.7%" width="18.5%"
                                    height="18.5%" zIndex="3"/>
      <region id="iconReg" left="87.5%" top="11.7%" width="8.45%"
                                    height="6.7%" zIndex="3"/>
      <region id="shoesReg" left="15%" top="60%" width="25%"
                                    height="25%" zIndex="3"/>
      <region id="formReg" left="56.25%" top="8.33%" width="38.75%"
                                    height="71.7%" zIndex="3"/>
      <region id="intReg" left="92.5%" top="91.7%" width="5.07%"
                                    height="6.51%" zIndex="3"/>
      <region id="chorinhoReg" left="2.5%" top="91.7%"
                        width="11.7%" height="6.51%" zIndex="3"/>
      <region id="rockReg" left="25%" top="91.7%" width="11.7%"
                                    height="6.51%" zIndex="3"/>
      <region id="technoReg" left="47.5%" top="91.7%" width="11.7%"
                                    height="6.51%" zIndex="3"/>
      <region id="cartoonReg" left="70%" top="91.7%" width="11.7%"
                                    height="6.51%" zIndex="3"/>
   </region>
</regionBase>
```

Listado 3.43 Base de regiones de la nueva versión.

Como próximo paso definimos el nuevo conjunto de descriptores. Deben ser especificados descriptores para cada uno de los cuatro íconos a ser presentados para la selección del ritmo.

En la definición de cada descriptor, debemos informar su índice para la navegación por las teclas de control remoto. Esto es hecho por el atributo *focusIndex* del elemento <descriptor>. Debemos también informar los próximos elementos a recibir el foco cuando naveguemos por medio de las teclas CURSOR_RIGHT y CURSOR_LEFT, indicando los próximos valores de *focusIndex* que deberán ser objeto de foco. Esto será realizado por medio de atributos *moveRight* y *moveLeft*, respectivamente. Así, tomando como ejemplo el elemento <descriptor> “chorinhoDesc” del Listado 3.44, vemos que su índice para foco es “2” y cuando el elemento <media> que el referencia esta con el foco y la tecla CURSOR_RIGHT del control remoto es presionada, el foco es movido para el elemento <media> qué referencia el elemento <descriptor> con el atributo *focusIndex* igual a “3”. Análogamente, cuando el elemento <media> que referencia el elemento <descriptor> con el atributo *focusIndex* “chorinhoDesc” esta con el foco y la tecla CURSOR_LEFT del control remoto es presionada el foco es movido para el elemento <media> que referencia el elemento <descriptor> con el atributo *focusIndex* igual a “5”. El Listado 3.44 ilustra la definición de los nuevos descriptores. Se puede observar que la navegación por los íconos que presentan los ritmos es circular.

```
<descriptorBase>

   <descriptor id="chorinhoDesc" region="chorinhoReg" focusIndex="2"
                                    moveRight="3" moveLeft="5"/>
   <descriptor id="rockDesc" region="rockReg" focusIndex="3"
                                    moveRight="4" moveLeft="2"/>
   <descriptor id="technoDesc" region="technoReg" focusIndex="4"
```

```
                                    moveRight="5" moveLeft="3"/>
   <descriptor id="cartoonDesc" region="cartoonReg" focusIndex="5"
                                    moveRight="2" moveLeft="4"/>
</descriptorBase>
```

Listado 3.44 Nuevos descriptores con la definición de atributos para navegación por teclas.

El lector ya debe haber percibido que además de los cuatro íconos mencionados, también tenemos que incrementar a la aplicación tres objetos de audio, considerando que el audio "chorinho" ya estaba presente en la versión anterior.

Vamos ahora a establecer el siguiente escenario para nuestra aplicación. Como anteriormente, el audio "chorinho" comenzó 5 segundos después del inicio de la aplicación. Justamente con el inicio del "choro", vamos también iniciar la presentación de los íconos (imágenes) para "chorinho", "rock", "techno" y "cartoon". Por lo tanto, vamos a colocar el elemento <media> "choro" del tipo audio e los elementos <media> "imgChorinho", "imgRock", "imgTechno" y "imgCartoon", del tipo imagen, dentro (como hijo) de un elemento <context> "menú". También vamos a colocar cinco elementos <port> cada uno mapeando para cada uno de los nuevos elementos <media> definidos. Realizar una acción "start" en un contexto sin especificar el puerto significa que todos los puertos deben recibir la acción, bastará tener una relación especificando el "start" del elemento <context> "menu" para que todos los íconos sean presentados y el audio "chorinho" iniciado, como queríamos.

Continuando con nuestro escenario vamos a establecer que en cualquier instante en que uno de los íconos sea accionado, con excepción del ícono del "chorinho", este tendrá su volumen colocado en cero "zero" (mudo), el ritmo correspondiente a el ícono seleccionado iniciará su presentación y los demás ritmos serán abruptamente terminados. En cualquier momento que el ícono del "chorinho" sea seleccionado, los demás ritmos cesarán y el audio del "chorinho" será restablecido en su valor inicial.

Para conseguir el efecto descrito en el párrafo anterior más fácilmente, vamos a reunir los tres audios (elementos <media> "rock", "techno" y "cartoon") en un elemento <switch>, identificados como "musics" y colocado también como hijo del elemento <context> "menú". El elemento <switch> tendrá un puerto (elemento <switchPort>) mapeado para los tres elementos de multimedia representando los audios, que serán escogidos conforme el valor del atributo *focusIndex* del elemento <media> representando el ícono asociado al ritmo que fue seleccionado por la tecla ENTER del control remoto. Por tanto, la variable global "service.currentFocus" (una propiedad *settings* ya discutido) será consultada.

Esa variable es mantenida por el sistema de la aplicación NCL, conteniendo siempre el valor del atributo *focusIndex* que se encuentra corriendo.

Cuando queramos parar todos los elementos hijos del elemento <context> “menú”, bastará realizar una acción “stop” sobre el contexto, sin especificar ningún puerto. En la nueva versión, vamos a definir que esto suceda cuando el video de animación llegue al punto en que son presentados los créditos de los autores (elementos <area> “segCred”). También vamos aprovechar para definir el término del contexto de interactividad para ese punto del video de la animación. El Listado 3.45 ilustra la definición del elemento <link> que permite hacer el procedimiento definido.

```
<link xconnector="conEx#onEndStop">
   <bind role="onEnd" component="animation" interface="segCred"/>
   <bind role="stop" component="menu"/>
   <bind role="stop" component="interactivity"/>
</link>
```

Listado 3.45 Finalización de todos los elementos en exhibición del contexto “menu”.

La Figura 3.18 preseta la visión estructural para esta nueva versión del primer *João* con navegación por teclas.

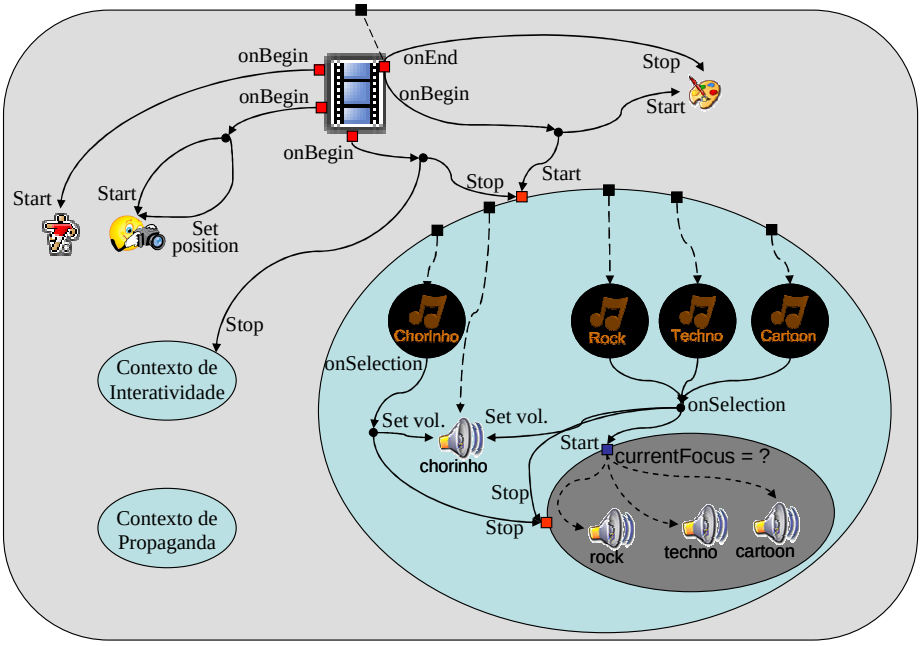


Figura 3.18 Visión estructural del primer *João* con navegación por teclas.

Se puede observar en la Figura 3.18 que la selección de cualquier de los tres íconos “imgRock”, “imgTechno” o “imgCartoon”, coloca el volumen del “chorinho” en cero, para la exhibición de los demás audios e

inicia la presentación del audio correspondiente a el foco correspondiente, es decir, al audio asociado al ícono seleccionado.

Para definir el elemento <switch> “menu”, nuevas reglas de selección deben ser definidas en el elemento <ruleBase> que en su nueva versión sigue la especificación ilustrada en el Listado 3.46. Note que las nuevas reglas introducidas dicen respecto a la variable “service.currentFocus”, que tendrá su valor testado contra el valor del atributo *focusIndex* de cada uno de los íconos asociado a los elementos <media > representando los varios ritmos. La regla es considerada satisfecha si hubiera igualdad entre los valores.

```
<ruleBase>
   <rule id="en" var="system.language" value="en" comparator="eq"/>
   <rule id="int" var="service.interactivity" value="true"
                                              comparator="eq"/>
   <rule id="rRock" var="service.currentFocus" value="3"
                                              comparator="eq"/>
   <rule id="rTechno" var="service.currentFocus" value="4"
                                              comparator="eq"/>
   <rule id="rCartoon" var="service.currentFocus" value="5"
                                              comparator="eq"/>
</ruleBase>
```

Listado 3.46 Redefinición del elemento <ruleBase> y sus nuevos elementos hijos.

Ahora podemos definir el elemento <context> “menu” incluyendo entre sus hijos el elemento <switch> “musics” para escoger los ritmos y los demás elementos discutidos en esta sección, conforme ilustra el Listado 3.47.

```
<context id="menu">
   <port id="pChoro" component="choro"/>
   <port id="pChorinho" component="imgChorinho"/>
   <port id="pRock" component="imgRock"/>
   <port id="pTechno" component="imgTechno"/>
   <port id="pCartoon" component="imgCartoon"/>
   <media id="imgChorinho" src="../media/chorinho.png"
                                    descriptor="chorinhoDesc"/>
   <media id="imgRock" src="../media/rock.png"
                                    descriptor="rockDesc"/>
   <media id="imgTechno" src="../media/techno.png"
                                    descriptor="technoDesc"/>
   <media id="imgCartoon" src="../media/cartoon.png"
                                    descriptor="cartoonDesc"/>
   <media id="choro" src="../media/choro.mp3" descriptor="audioDesc">
      <property name="soundLevel" value="1"/>
   </media>
   <switch id="musics">
      <bindRule constituent="rock" rule="rRock"/>
      <bindRule constituent="techno" rule="rTechno"/>
      <bindRule constituent="cartoon" rule="rCartoon"/>
      <media id="rock" src="../media/rock.mp3"/>
      <media id="techno" src="../media/techno.mp3"/>
         <media id="cartoon" src="../media/cartoon.mp3"/>
```

```
   </switch>
   <link id="lChoro" xconnector="conEx#onSelectionSetStop">
      <bind role="onSelection" component="imgChorinho"/>
      <bind role="set" component="choro" interface="soundLevel">
         <bindParam name="var" value="1"/>
      </bind>
      <bind role="stop" component="musics"/>
   </link>
   <link id="lOthers" xconnector="conEx#onOrSelectionSetStopStart">
      <bind role="onSelection" component="imgRock"/>
      <bind role="onSelection" component="imgTechno"/>
      <bind role="onSelection" component="imgCartoon"/>
      <bind role="set" component="choro" interface="soundLevel">
         <bindParam name="var" value="0"/>
      </bind>
      <bind role="stop" component="musics"/>
      <bind role="start" component="musics"/>
   </link>
</context>
```

Listado 3.47 Elemento <context> “menu”.

Note, en el Listado 3.47, la definición de los puertos para todos los íconos que representan los diferentes ritmos y para el audio del “chorinho”, conforme discutimos anteriormente. Note también la definición del elemento <switch> “musics” y además la definición de los elementos <link>. La primera relación (“lChoro”) especifica que, al ser seleccionada (condición “onSelection”) la imagen correspondiente a los íconos del chorinho, el sonido del audio del choro debe ser colocado en su volumen original (acción “set” colocando el valor de *soundLevel* en “1”), y todos los otros ritmos deben ser encerados (acción de “stop” en el elemento <switch>”musics”). La segunda relación (“lOthers”) especifica que al ser seleccionada (condición “onSelection”) cualquier de las imágenes correspondientes a los otros ritmos (rock, techno y cartoon), el volumen del audio del choro debe ser bajado totalmente (acción “set” colocando el valor de *soundLevel* en “0”), cualquier otro ritmo que estuviera siendo tocado debe parar (acción de “stop” en el elemento <switch> “musics”), y el audio correspondiente a el ícono seleccionado (cuyo valor *focusIndex* es entonces atribuido a la propiedad “service.currentFocus”) debe ser iniciado (acción de “start” en el elemento <switch> “musics”, con la selección del componente segundo el valor de la variable “service.currentFocus”).

Finalmente, podemos realizar las dos últimas alteraciones de la versión anterior de la aplicación. La primera alterando el elemento <link> “lMusic” para que el ahora inicie la presentación del elemento <context> “menú” a partir de los 5 segundos del inicio del video de la animación, esto es inicie la presentación del chorinho y de los íconos correspondientes a todos los ritmos, conforme ilustra el Listado 3.48.

```
<link id="lMusic" xconnector="conEx#onBeginStartDelay">
       <bind role="onBegin" component="animation"/>
```

```
        <bind role="start" component="background">
                <bindParam name="delay" value="5s"/>
        </bind>
        <bind role="start" component="menu">
                <bindParam name="delay" value="5s"/>
        </bind>
</link>
```

Listado 3.48 Redefinición del elemento <link> “lMusic”.

La segunda alteración es bien sutil. En la presentación del formulario para la compra del zapato, debemos obligatoriamente colocar el foco en el formulario, independientemente de donde el foco se encuentre (en relación a los íconos representan los diversos ritmos). Esa acción tornará posible el llenado del formulario. Caso no lo hagamos, el foco quedará circulando entre los íconos y no será colocado en el formulario. Note que cuando el formulario gana el foco, él solo será nuevamente pasado a los íconos en el final de su presentación. El Listado 3.49 ilustra el elemento <link> responsable por determinar el foco en el formulario, así que se inicie su presentación. (note que para establecer el foco basta atribuir la variable global “service.currentFocus” el valor del atributo *focusIndex* del formulario (en el caso el valor “1”).

```
<link id="lFormFocus" xconnector="conEx#onBeginSet">
        <bind role="onBegin" component="form"/>
        <bind role="set" component="reusedGlobalVar"
                                interface="service.currentFocus">
                <bindParam name="var" value="1"/>
        </bind>
</link>
```

Lista 3.49 Pasando el foco al formulario cuando inicia su presentación.

La especificación completa de la nueva versión del programa NCL es ilustrada en el Listado 3.50.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Ejemplo de navegación por menu -->
<ncl id="menuEx" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
   <head>
      <ruleBase>
         <rule id="en" var="system.language" value="en"
                                                comparator="eq"/>
         <rule id="int" var="service.interactivity" value="true"
                                                comparator="eq"/>
         <rule       id="rRock"       var="service.currentFocus"       value="3"
                                                comparator="eq"/>
         <rule id="rTechno" var="service.currentFocus" value="4"
                                                comparator="eq"/>
         <rule id="rCartoon" var="service.currentFocus" value="5"
                                                comparator="eq"/>
      </ruleBase>
      <transitionBase>
         <transition id="trans1" type="fade" dur="2s"/>
         <transition id="trans2" type="barWipe" dur="1s"/>
```

```
        </transitionBase>
        <regionBase>
            <region id="backgroundReg" width="100%" height="100%"
                                                          zIndex="1">
                <region id="screenReg" width="100%" height="88%"
                                                          zIndex="2"/>
                <region id="frameReg" left="5%" top="6.7%" width="18.5%"
                                          height="18.5%" zIndex="3"/>
                <region id="iconReg" left="87.5%" top="11.7%"
                          width="8.45%" height="6.7%" zIndex="3"/>
                <region id="shoesReg" left="15%" top="60%" width="25%"
                                          height="25%" zIndex="3"/>
                <region id="formReg" left="56.25%" top="8.33%"
                          width="38.75%" height="71.7%" zIndex="3"/>
                <region id="intReg" left="92.5%" top="91.7%"
                          width="5.07%" height="6.51%" zIndex="3"/>
                <region id="chorinhoReg" left="2.5%" top="91.7%"
                          width="11.7%" height="6.51%" zIndex="3"/>
                <region id="rockReg" left="25%" top="91.7%"
                          width="11.7%" height="6.51%" zIndex="3"/>
                <region id="technoReg" left="47.5%" top="91.7%"
                          width="11.7%" height="6.51%" zIndex="3"/>
                <region id="cartoonReg" left="70%" top="91.7%"
                          width="11.7%" height="6.51%" zIndex="3"/>
            </region>
        </regionBase>

        <descriptorBase>
            <descriptor id="backgroundDesc" region="backgroundReg"/>
            <descriptor id="screenDesc" region="screenReg"/>
            <descriptor id="photoDesc" region="frameReg"
                                                    explicitDur="5s">
                <descriptorParam name="transparency" value="0.6"/>
            </descriptor>
            <descriptor id="audioDesc"/>
            <descriptor id="dribleDesc" region="frameReg"
                                  transIn="trans1" transOut="trans2"/>
            <descriptor id="iconDesc" region="iconReg"
                                                    explicitDur="6s"/>
            <descriptor id="shoesDesc" region="shoesReg"/>
            <descriptor id="formDesc" region="formReg" focusIndex="1"
                                                    explicitDur="15s"/>
            <descriptor id="intDesc" region="intReg"/>
            <descriptor id="chorinhoDesc" region="chorinhoReg"
                          focusIndex="2" moveRight="3" moveLeft="5"/>
            <descriptor id="rockDesc" region="rockReg" focusIndex="3"
                                          moveRight="4" moveLeft="2"/>
            <descriptor id="technoDesc" region="technoReg"
                          focusIndex="4" moveRight="5" moveLeft="3"/>
            <descriptor id="cartoonDesc" region="cartoonReg"
                          focusIndex="5" moveRight="2" moveLeft="4"/>
        </descriptorBase>
        <connectorBase>
            <importBase documentURI="../causalConnBase.ncl"
                                                          alias="conEx"/>
        </connectorBase>
    </head>
    <body>
        <port id="entry" component="animation"/>
        <media id="background" src="../media/background.png"
                                          descriptor="backgroundDesc"/>
        <media id="animation" src="../media/animGar.mp4"
                                          descriptor="screenDesc">
            <area id="segDrible" begin="12s"/>
            <area id="segPhoto" begin="41s"/>
            <area id="segIcon" begin="45s" end="51s"/>
            <area id="segCred" end="62s"/>
        </media>
        <media id="drible" src="../media/drible.mp4"
                                          descriptor="dribleDesc"/>
        <media id="photo" src="../media/photo.png"
                                          descriptor="photoDesc">
```

```
            <property name="top"/>
        </media>
        <context id="interactivity">
            <media id="globalVar" type="application/x-ncl-settings">
                <property name="service.interactivity" value="true"/>
                <property name="service.currentFocus"/>
            </media>
            <media id="anotherAnimation" refer="animation"
                                                    instance="instSame"/>
            <media id="intOn" src="../media/intOn.png"
                                                    descriptor="intDesc"/>
            <media id="intOff" src="../media/intOff.png"
                                                    descriptor="intDesc"/>
            <link id="lInt" xconnector="conEx#onBeginSetStart">
                <bind role="onBegin" component="anotherAnimation"/>
                <bind role="start" component="intOn"/>
                <bind role="set" component="globalVar"
                                    interface="service.interactivity">
                  <bindParam name="var" value="true"/>
                </bind>
            </link>
            <link id="lOn"
                   xconnector="conEx#onKeySelectionStopSetStart">
                <bind role="onSelection" component="intOn">
                  <bindParam name="keyCode" value="INFO"/>
                </bind>
                <bind role="start" component="intOff"/>
                <bind role="stop" component="intOn"/>
                <bind role="set" component="globalVar"
                                    interface="service.interactivity">
                  <bindParam name="varSet" value="false"/>
                </bind>
            </link>
            <link id="lOff"
                   xconnector="conEx#onKeySelectionStopSetStart">
                <bind role="onSelection" component="intOff">
                  <bindParam name="keyCode" value="INFO"/>
                </bind>
                <bind role="start" component="intOn"/>
                <bind role="stop" component="intOff"/>
                <bind role="set" component="globalVar"
                                    interface="service.interactivity">
                  <bindParam name="varSet" value="true"/>
                </bind>
            </link>
        </context>
        <context id="advert">
            <media id="reusedAnimation" refer="animation"
                                                    instance="instSame">
                <property name="bounds"/>
            </media>
            <media id="reusedGlobalVar" refer="globalVar"
                                                    instance="instSame"/>
            <media id="icon" src="../media/icon.png"
                                                descriptor="iconDesc"/>
            <media id="shoes" src="../media/shoes.mp4"
                                                descriptor="shoesDesc"/>
            <switch id="form">
                <switchPort id="spForm">
                  <mapping component="enForm"/>
                  <mapping component="ptForm"/>
                </switchPort>
                <bindRule constituent="enForm" rule="en"/>
                <defaultComponent component="ptForm"/>
                <media id="ptForm" src="../media/ptForm.htm"
                            type="text/html" descriptor="formDesc"/>
                <media id="enForm" src="../media/enForm.htm"
                            type="text/html" descriptor="formDesc"/>
            </switch>
            <link id="lIcon" xconnector="conEx#onBeginVarStart">
```

```
            <bind role="onBegin" component="reusedAnimation"
                                                interface="segIcon"/>
            <bind role="var" component="reusedGlobalVar"
                                interface="service.interactivity"/>
            <bind role="start" component="icon"/>
         </link>
         <link id="lBeginShoes"
               xconnector="conEx#onKeySelectionStopSetStart">
            <bind role="onSelection" component="icon">
              <bindParam name="keyCode" value="RED"/>
            </bind>
            <bind role="start" component="shoes"/>
            <bind role="start" component="form" interface="spForm"/>
            <bind role="set" component="reusedAnimation"
                                                interface="bounds">
              <bindParam name="varSet" value="5%,6.67%,45%,45%"/>
            </bind>
            <bind role="stop" component="icon"/>
         </link>
         <link id="lFormFocus" xconnector="conEx#onBeginSet">
            <bind role="onBegin" component="form"/>
            <bind role="set" component="reusedGlobalVar"
                                interface="service.currentFocus">
              <bindParam name="var" value="1"/>
            </bind>
         </link>
         <link id="lEndForm" xconnector="conEx#onEndSet">
            <bind role="onEnd" component="form" interface="spForm"/>
            <bind role="set" component="reusedAnimation"
                                                interface="bounds">
              <bindParam name="varSet" value="0,0,222.22%,222.22%"/>
            </bind>
         </link>
      </context>
      <context id="menu">
         <port id="pChoro" component="choro"/>
         <port id="pChorinho" component="imgChorinho"/>
         <port id="pRock" component="imgRock"/>
         <port id="pTechno" component="imgTechno"/>
         <port id="pCartoon" component="imgCartoon"/>
         <media id="imgChorinho" src="../media/chorinho.png"
                                            descriptor="chorinhoDesc"/>
         <media id="imgRock" src="../media/rock.png"
                                            descriptor="rockDesc"/>
         <media id="imgTechno" src="../media/techno.png"
                                            descriptor="technoDesc"/>
         <media id="imgCartoon" src="../media/cartoon.png"
                                            descriptor="cartoonDesc"/>
         <media id="choro" src="../media/choro.mp3"
                                            descriptor="audioDesc">
            <property name="soundLevel" value="1"/>
         </media>
         <switch id="musics">
            <bindRule constituent="rock" rule="rRock"/>
            <bindRule constituent="techno" rule="rTechno"/>
            <bindRule constituent="cartoon" rule="rCartoon"/>
            <media id="rock" src="../media/rock.mp3"/>
            <media id="techno" src="../media/techno.mp3"/>
            <media id="cartoon" src="../media/cartoon.mp3"/>
         </switch>
         <link id="lChoro" xconnector="conEx#onSelectionSetStop">
            <bind role="onSelection" component="imgChorinho"/>
            <bind role="set" component="choro"
                                            interface="soundLevel">
              <bindParam name="var" value="1"/>
            </bind>
            <bind role="stop" component="musics"/>
         </link>
         <link id="lOthers"
                        xconnector="conEx#onOrSelectionSetStopStart">
            <bind role="onSelection" component="imgRock"/>
            <bind role="onSelection" component="imgTechno"/>
```

```
            <bind role="onSelection" component="imgCartoon"/>
            <bind role="set" component="choro"
                                        interface="soundLevel">
              <bindParam name="var" value="0"/>
            </bind>
            <bind role="stop" component="musics"/>
            <bind role="start" component="musics"/>
         </link>
      </context>
      <link id="lMusic" xconnector="conEx#onBeginStartDelay">
         <bind role="onBegin" component="animation"/>
         <bind role="start" component="background">
            <bindParam name="delay" value="5s"/>
         </bind>
         <bind role="start" component="menu">
            <bindParam name="delay" value="5s"/>
         </bind>
      </link>
      <link id="lDrible" xconnector="conEx#onBeginStart">
         <bind role="onBegin" component="animation"
                                        interface="segDrible"/>
         <bind role="start" component="drible"/>
      </link>
      <link id="lPhoto" xconnector="conEx#onBeginStartSetDelay">
         <bind role="onBegin" component="animation"
                                              interface="segPhoto"/>
         <bind role="start" component="photo"/>
         <bind role="set" component="photo" interface="top">
            <bindParam name="var" value="290"/>
            <bindParam name="delay" value="1s"/>
            <bindParam name="duration" value="3s"/>
         </bind>
      </link>
      <link xconnector="conEx#onEndStop">
         <bind role="onEnd" component="animation"
                                              interface="segCred"/>
         <bind role="stop" component="menu"/>
         <bind role="stop" component="interactivity"/>
      </link>
      <link id="lEnd" xconnector="conEx#onEndStop">
         <bind role="onEnd" component="animation"/>
         <bind role="stop" component="background"/>
      </link>
   </body>
</ncl>
```

Listado 3.50 El primer *João* con navegación del menú por teclas.

En la Figura 3.19 se ilustra la navegación por teclas en la implementación de referencia del middleware Ginga-NCL. En la parte izquierda de la figura, el foco se encuentra en el ícono de selección de la música llamada *chorinho*. En la derecha, el foco se desplaza para el ícono que selecciona la música llamada *techno*.

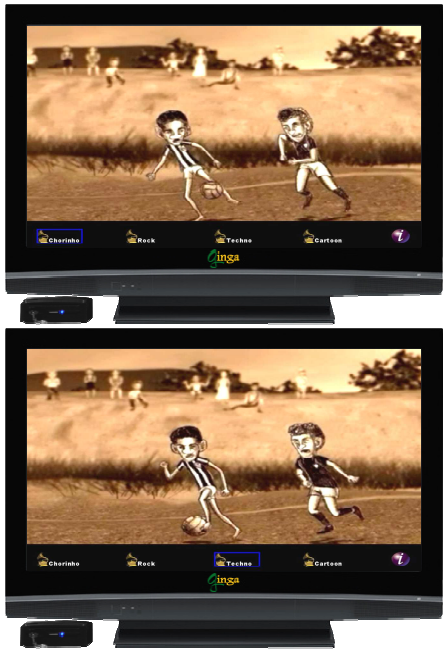


Figura 3.19 Escenas de la aplicación del primer *João* con navegación por el menú con teclas.

3.12 Incrementado un Objeto NCLua

La última funcionalidad que agregaremos a nuestro ejemplo será un nodo con código *Lua*. Nuestro objetivo es implementar un contador que almacenará cuantas veces el usuario de nuestro ejemplo del primer *João* cambió de ritmo musical. En otras palabras, cada vez que un nuevo ritmo es escogido, el contador es incrementado. Al final de la presentación del video de la animación, el resultado será presentado en la pantalla: "*Número de veces que cambió el ritmo: x.*"

Para realizar esta programación, vamos a crear un objeto NCLua (elemento <media> del tipo "application/x-ncl-NCLua), que identificaremos por "changes", el mismo que contiene un script *Lua*, que al ser inicializado, colocará su variable interna *counter* con el valor de "0". También vamos a definir una propiedad para este objeto de multimedia denominada "inc". Cada vez que un valor numérico fuera atribuido a esta propiedad, por medio de un elemento <link>, el objeto NCLua sumará este valor a la variable *counter*.

La especificación del objeto de multimedia NCLua es dada por el elemento <media> "changes" en el Listado 3.51, cuyo contenido se encuentra almacenado en el archivo "counter.lua". El lector puede notar que el tipo de objeto (atributo *type*) no requiere ser declarado, porque el

interprete NCL lo infiere a partir de la extensión ".lua" del archivo impuesto al atributo *source* del elemento.

En el Listado 3.51, definimos el elemento <media> "changes" como hijo del elemento <context> "menu", donde están también todos los íconos para selección de los ritmos. A partir de ahora, cuando el elemento <context> "menu" sea inicializado, no solo los íconos correspondientes a los ritmos serán exhibidos junto con el audio *chorinho*, sino también se inicializará el objeto NCLua, conforme lo especificado por todos los puertos del elemento <contex> "menu" que están mapeados para ser vinculados a los nodos hijos del contexto.

Note que un nuevo puerto fue definido en el elemento <context> "menu", que mapea para la propiedad "inc" del objeto NCLua. Cuando a esta propiedad se le atribuya el valor "FIM", el valor de variable *counter* interna al objeto NCLua será exhibida en la frase "Número de veces que cambió de ritmo: *valor de la variable counter*" al final del video de la animación, antes de los créditos.

```
<context id="menu">
        <port id="pChoro" component="choro"/>
        <port id="pChorinho" component="imgChorinho"/>
        <port id="pRock" component="imgRock"/>
        <port id="pTechno" component="imgTechno"/>
        <port id="pCartoon" component="imgCartoon"/>
        <port id="pNCLua" component="changes"/>
        <port id="pChanges" component="changes" interface="inc"/>
...
        <media id="changes" src="../script/counter.lua"
                                        descriptor="changesDesc">
            <property name="inc"/>
        </media>
...
</context>
```

Listado 3.51 Objeto de multimedia NCLua.

El resultado de la ejecución del script será presentado en una región de la pantalla referenciada en el elemento <descriptor> y definida en el elemento <region>, conforme se ilustra en el Listado 3.52.

```
<regionBase>
   <region id="backgroundReg" width="100%" height="100%" zIndex="1">
...
       <region id="changesReg" left="0%" top="90%" width="100%"
                                       height="10%" zIndex="4"/>
   </region>
</regionBase>
<descriptorBase>
...
   <descriptor id="changesDesc" region="changesReg"/>
</descriptorBase>
```

Listado 3.52 Definición de la región de presentación del objeto NCLua.

Para incrementar la variable *counter* del script *Lua* cada vez que un ritmo es cambiado, se atribuye el valor "1" a la propiedad "inc" del objeto NCLua. Haremos esto incrementando un elemento <bind> al enlace de la selección de los ritmos "rock", "techno", "cartoon" y otro al enlace de la selección del ícono "chorinho". Estos enlaces, pertenecientes al elemento <context> "menu" quedan como se ilustra en el Listado 3.53.

```
<context id="menu">
...
       <link id="lChoro" xconnector="conEx#onSelectionSetStop">
          <bind role="onSelection" component="imgChorinho"/>
          <bind role="set" component="choro" interface="soundLevel">
               <bindParam name="var" value="1"/>
          </bind>
          <bind role="set" component="changes" interface="inc">
               <bindParam name="var" value="1"/>
          </bind>
          <bind role="stop" component="musics"/>
       </link>
       <link id="lOthers"
                     xconnector="conEx#onOrSelectionSetStopStart">
          <bind role="onSelection" component="imgRock"/>
          <bind role="onSelection" component="imgTechno"/>
          <bind role="onSelection" component="imgCartoon"/>
          <bind role="set" component="choro" interface="soundLevel">
               <bindParam name="var" value="0"/>
          </bind>
          <bind role="set" component="changes" interface="inc">
               <bindParam name="var" value="1"/>
          </bind>
          <bind role="stop" component="musics"/>
          <bind role="start" component="musics"/>
       </link>
</context>
```

Listado 3.53 Enlaces para incrementar la variable *counter*.

Como ya mencionamos, el objeto NCLua será inicializado conjuntamente con el elemento <context> “menu”. Para finalizar el objeto vamos a modificar el elemento <link> que determina el final de la exhibición de todos los componentes del elemento <contexto> “menu”, que ahora también incluye el objeto NCLua. El nuevo enlace, ilustrado en el Listado 3.54, determina que al final del video de la animación, cuando se inicien los créditos, será exhibido, durante 3 segundos, el número de veces que los ritmos fueron cambiados. Luego de este tiempo, la exhibición de todos los hijos del elemento <context> “menu” será terminada, como también la exhibición de los elementos hijos del contexto que determina la interactividad.

```
<link xconnector="conEx#onEndSetStopDelay">
        <bind role="onEnd" component="animation" interface="segCred"/>
        <bind role="set" component="menu" interface="pChanges">
                <bindParam name="var" value="FIM"/>
        </bind>
        <bind role="stop" component="menu"/>
        <bind role="stop" component="interactivity"/>
</link>
```

Listado 3.54 Enlace para la exhibición del resultado final de los cambios del ritmo y para terminar la exhibición de contextos que definen la navegación por teclas y la interactividad.

El lector puede notar que el nuevo enlace creado hace referencia al nuevo elemento <causalConnector>, definido en el Listado 3.55 (vea también el Apéndice D). El conector que determina el retardo de 3 segundos debe ser asociado a la acción “stop”. El retardo no fue parametrizado con el propósito de ejemplificar el uso de una acción parametrizada (“set”) y de una acción no parametrizada.

```
<causalConnector id="onEndSetStopDelay">
        <connectorParam name="var"/>
        <simpleCondition role="onEnd"/>
        <compoundAction operator="seq">
            <simpleAction role="set" value="$var"/>
            <simpleAction role="stop" max="unbounded"
                                      qualifier="par" delay="3s"/>
        </compoundAction>
</causalConnector>
```

Listado 3.55 Nuevo elemento <causalConnector> para el enlace del Código Programa 3.55.

La Figura 3.20 presenta la visión estructural de la versión final del ejemplo del primer *João*.

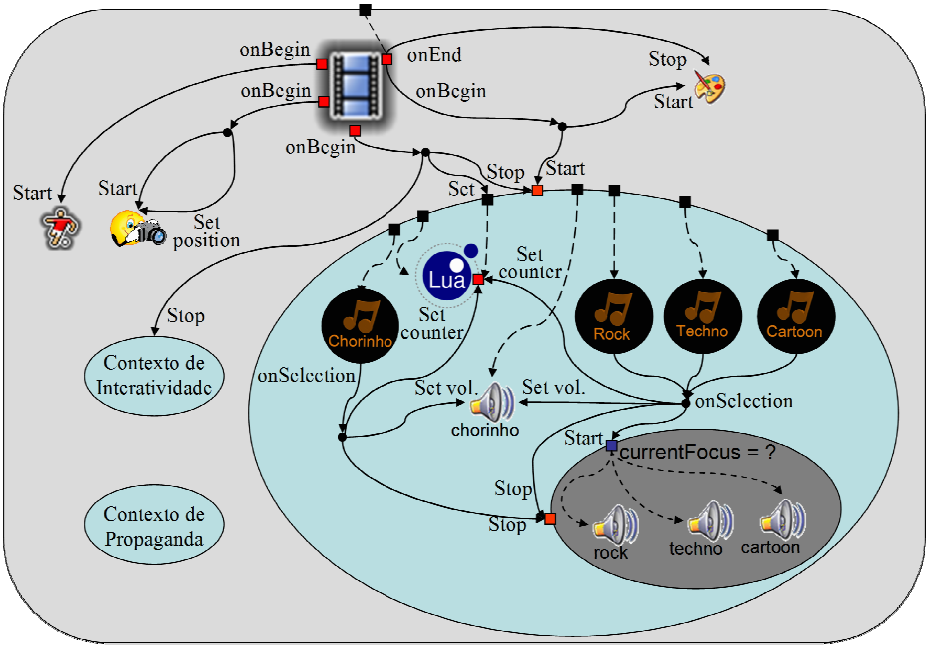


Figura 3.20 Visión estructural de la versión final del ejemplo El primer *João*.

Ahora vamos a ver el contenido del objeto NCLua. Al inicializar un objeto con código *Lua*, no importa por cual de sus vínculos, el objeto primero ejecuta una rutina de inicialización, cuando todas las funciones manejadoras de eventos disparados interna o externamente al objeto son registradas. Para nuestro ejemplo, queremos registrar una función que manejará eventos que cambian el valor de la propiedad “inc” del elemento <media> “changes”. El Listado 3.56 muestra el script Lua de nuestro ejemplo.

En el Listado 3.56, en la primera línea se inicializa el contador (variable local *counter*) en cero. En la segunda línea se definen las variables que contienen el ancho y alto de la región “canvas”, donde será presentado el resultado. Estos valores son obtenidos por la función `canvas:attrSize()`, que los obtiene del elemento <descriptor> del objeto NCLua: en nuestro caso (vea el Listado 3.52) “100%” y “10%” de la pantalla de exhibición, respectivamente. De la línea 3 hasta la 23, tenemos la definición de la función que manejará el evento de atribución de valores a la propiedad “inc” del objeto NCLua. La línea 24 registra esta función, es decir, deja el objeto listo para recibir eventos que llamarán a la función manejadora. Todo el procedimiento descrito en este párrafo es ejecutado cuando el objeto NCLua es inicializado por la acción “start” sobre el elemento <context> “menu”.

```
1    local counter = 0
2    local dx, dy = canvas:attrSize()        -- dimensiones de canvas
```

```
3   function handler (evt)
4       if evt.class =='ncl' and evt.type=='attribution' and
                                              evt.name=='inc' then
5           if evt.value ~= 'FIM' then
6              counter = counter + evt.value
7           else
8              canvas:attrColor ('black')
9              canvas:drawRect('fill',0,0,dx,dy)
10             canvas:attrColor ('yellow')
11             canvas:attrFont ('vera', 24, 'bold')
12             canvas:drawText (10,10, 'Número de veces que cambió
                                             de ritmo: '..counter)
13             canvas:flush()
14          end
15          event.post {
16             class   = 'ncl',
17             type    = 'attribution',
18             name    = 'inc',
19             action  = 'stop',
20             value   = counter,
21          }
22      end
23  end
24  event.register(handler)
```

Listado 3.56 Script Lua del elemento <media> “changes”.

Como se indicó en la línea 4 del Listado 3.56, la función manejadora de evento es llamada siempre que se inicia (note bien que es el inicio, ya que retornaremos a este punto más adelante en este texto) la atribución al elemento <propriedade> cuyo atributo *name*=“inc”, y esta acción de atribución parte de un enlace del documento NCL (evt.class= = 'ncl').

Las líneas 5 y 6 indican que en caso que el valor atribuido a la propiedad “inc” no sea “FIM”, el valor de la variable *counter* debe ser adicionado al nuevo valor atribuido. Como en nuestro caso, el valor atribuido es siempre “1” (vea el Listado 3.53), el valor de la variable *counter* es incrementado.

Las líneas 7 a 14 indican lo que sucede en el caso que el valor atribuido a la propiedad “inc” sea “FIM”. En este caso, queremos imprimir en la pantalla el valor final de la variable *counter*. Por lo tanto, las líneas 8 y 9 comandan el llenado de toda la región “canvas” con el color negro. Las líneas 10 y 11 establecen el color y la fuente para el texto del mensaje. La línea 12 comanda la impresión del mensaje “*Número de veces que cambió de ritmo:*”, concatenado con el valor de la variable *counter*. La línea 13 comanda la actualización (refrescando) de la región “canvas”.

La acción “start” sobre la propiedad “inc” que proviene de un enlace del documento NCL, se inicia apenas se atribuye un valor a esta propiedad. Independientemente del valor que sea atribuido, el final de la atribución debe ser señalizado por el objeto NCLua, tan pronto se termine la realización de las tareas llamadas por el inicio del evento de atribución. Esto es realizado por las líneas15 a 20, en el Listado 3.56. Estas líneas comandan la generación de un evento (`event.post`) de fin de atribución en la propiedad “inc”, dejando como valor final de esta propiedad el valor de la variable local *counter*. Note que la atribución comienza atribuyendo el valor “1” (que puede ser visto como parámetro de entrada para la función manejadora) y termina con el valor de la variable *counter* (que puede ser visto como parámetro de salida de la función manejadora).

La especificación completa de la nueva versión del programa NCL es ilustrada en el Listado 3.57.

```
<?xml version=”1.0” encoding=”ISO-8859-1"?>
<!-- Ejemplo de uso de objeto NCLua -->
<ncl id="nclua" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
   <head>
      <ruleBase>
         <rule id="en" var="system.language" value="en"
                                                comparator="eq"/>
         <rule id="int" var="service.interactivity" value="true"
                                                comparator="eq"/>
         <rule id="rRock" var="service.currentFocus" value="3"
                                                comparator="eq"/>
         <rule id="rTechno" var="service.currentFocus" value="4"
                                                comparator="eq"/>
         <rule id="rCartoon" var="service.currentFocus" value="5"
                                                comparator="eq"/>
      </ruleBase>
      <transitionBase>
         <transition id="trans1" type="fade" dur="2s"/>
         <transition id="trans2" type="barWipe" dur="1s"/>
      </transitionBase>
      <regionBase>
         <region id="backgroundReg" width="100%" height="100%"
                                                zIndex="1">
           <region id="screenReg" width="100%" height="88%"
                                                    zIndex="2"/>
           <region id="frameReg" left="5%" top="6.7%" width="18.5%"
                                 height="18.5%" zIndex="3"/>
           <region id="iconReg" left="87.5%" top="11.7%" width="8.45%"
                                 height="6.7%" zIndex="3"/>
           <region id="shoesReg" left="15%" top="60%" width="25%"
                                 height="25%" zIndex="3"/>
           <region id="formReg" left="56.25%" top="8.33%"
                width="38.75%" height="71.7%" zIndex="3"/>
           <region id="intReg" left="92.5%" top="91.7%" width="5.07%"
                                 height="6.51%" zIndex="3"/>
           <region id="chorinhoReg" left="2.5%" top="91.7%"
                        width="11.7%" height="6.51%" zIndex="3"/>
           <region id="rockReg" left="25%" top="91.7%" width="11.7%"
                                 height="6.51%" zIndex="3"/>
           <region id="technoReg" left="47.5%" top="91.7%"
                        width="11.7%" height="6.51%" zIndex="3"/>
           <region id="cartoonReg" left="70%" top="91.7%" width="11.7%"
                                 height="6.51%" zIndex="3"/>
           <region id="changesReg" left="0%" top="90%" width="100%"
                                 height="10%" zIndex="4"/>
         </region>
```

```
        </regionBase>
        <descriptorBase>
            <descriptor id="backgroundDesc" region="backgroundReg"/>
            <descriptor id="screenDesc" region="screenReg"/>
            <descriptor id="photoDesc" region="frameReg" explicitDur="5s">
              <descriptorParam name="transparency" value="0.6"/>
            </descriptor>
            <descriptor id="audioDesc"/>
            <descriptor id="dribleDesc" region="frameReg" transIn="trans1"
                                          transOut="trans2"/>
            <descriptor id="iconDesc" region="iconReg" explicitDur="6s"/>
            <descriptor id="shoesDesc" region="shoesReg"/>
            <descriptor id="formDesc" region="formReg" focusIndex="1"
                                          explicitDur="15s"/>
            <descriptor id="intDesc" region="intReg"/>
            <descriptor id="chorinhoDesc" region="chorinhoReg"
                focusIndex="2" moveRight="3" moveLeft="5"/>
            <descriptor id="rockDesc" region="rockReg" focusIndex="3"
                                  moveRight="4" moveLeft="2"/>
            <descriptor id="technoDesc" region="technoReg" focusIndex="4"
                                  moveRight="5" moveLeft="3"/>
            <descriptor id="cartoonDesc" region="cartoonReg"
                  focusIndex="5" moveRight="2" moveLeft="4"/>
            <descriptor id="changesDesc" region="changesReg"/>
        </descriptorBase>
        <connectorBase>
            <importBase documentURI="../causalConnBase.ncl"
                                                    alias="conEx"/>
        </connectorBase>
    </head>

    <body>
        <port id="entry" component="animation"/>
        <media id="background" src="../media/background.png"
                                  descriptor="backgroundDesc"/>
        <media id="animation" src="../media/animGar.mp4"
                                  descriptor="screenDesc">
            <area id="segDrible" begin="12s"/>
            <area id="segPhoto" begin="41s"/>
            <area id="segIcon" begin="45s" end="51s"/>
            <area id="segCred" end="62s"/>
        </media>
        <media id="drible" src="../media/drible.mp4"
                                          descriptor="dribleDesc"/>
        <media id="photo" src="../media/photo.png"
                                  descriptor="photoDesc">
            <property name="top"/>
        </media>
        <context id="interactivity">
            <media id="globalVar" type="application/x-ncl-settings">
              <property name="service.interactivity" value="true"/>
              <property name="service.currentFocus"/>
            </media>
            <media id="anotherAnimation" refer="animation"
                                          instance="instSame"/>
            <media id="intOn" src="../media/intOn.png"
                                                    descriptor="intDesc"/>
            <media id="intOff" src="../media/intOff.png"
                                          descriptor="intDesc"/>
            <link id="lInt" xconnector="conEx#onBeginSetStart">
              <bind role="onBegin" component="anotherAnimation"/>
              <bind role="start" component="intOn"/>
              <bind role="set" component="globalVar"
                           interface="service.interactivity">
                    <bindParam name="var" value="true"/>
              </bind>
            </link>
            <link id="lOn" xconnector="conEx#onKeySelectionStopSetStart">
              <bind role="onSelection" component="intOn">
                    <bindParam name="keyCode" value="INFO"/>
```

```
            </bind>
            <bind role="start" component="intOff"/>
            <bind role="stop" component="intOn"/>
            <bind role="set" component="globalVar"
                        interface="service.interactivity">
                 <bindParam name="varSet" value="false"/>
            </bind>
          </link>
          <link id="lOff" xconnector="conEx#onKeySelectionStopSetStart">
            <bind role="onSelection" component="intOff">
                 <bindParam name="keyCode" value="INFO"/>
            </bind>
            <bind role="start" component="intOn"/>
            <bind role="stop" component="intOff"/>
            <bind role="set" component="globalVar"
                        interface="service.interactivity">
                 <bindParam name="varSet" value="true"/>
            </bind>
          </link>
        </context>
        <context id="advert">
          <media id="reusedAnimation" refer="animation"
                                          instance="instSame">
            <property name="bounds"/>
          </media>
          <media id="reusedGlobalVar" refer="globalVar"
                                          instance="instSame"/>
          <media id="icon" src="../media/icon.png"
                                          descriptor="iconDesc"/>
          <media id="shoes" src="../media/shoes.mp4"
                                              descriptor="shoesDesc"/>
          <switch id="form">
            <switchPort id="spForm">
                 <mapping component="enForm"/>
                 <mapping component="ptForm"/>
            </switchPort>
            <bindRule constituent="enForm" rule="en"/>
            <defaultComponent component="ptForm"/>
            <media id="ptForm" src="../media/ptForm.htm"
                        type="text/html" descriptor="formDesc"/>
            <media id="enForm" src="../media/enForm.htm"
                        type="text/html" descriptor="formDesc"/>
          </switch>
          <link  id="lIcon" xconnector="conEx#onBeginVarStart">
            <bind role="onBegin" component="reusedAnimation"
                                              interface="segIcon"/>
            <bind role="var" component="reusedGlobalVar"
                        interface="service.interactivity"/>
            <bind role="start" component="icon"/>
          </link>
          <link  id="lBeginShoes"
                 xconnector="conEx#onKeySelectionStopSetStart">
            <bind role="onSelection" component="icon">
                 <bindParam name="keyCode" value="RED"/>
            </bind>
            <bind role="start" component="shoes"/>
            <bind role="start" component="form" interface="spForm"/>
            <bind role="set" component="reusedAnimation"
                                          interface="bounds">
                 <bindParam name="varSet" value="5%,6.67%,45%,45%"/>
            </bind>
            <bind role="stop" component="icon"/>
          </link>
          <link id="lFormFocus" xconnector="conEx#onBeginSet">
            <bind role="onBegin" component="form"/>
            <bind role="set" component="reusedGlobalVar"
                                 interface="service.currentFocus">
                 <bindParam name="var" value="1"/>
            </bind>
          </link>
          <link id="lEndForm" xconnector="conEx#onEndSet">
            <bind role="onEnd" component="form" interface="spForm"/>
```

```
            <bind role="set" component="reusedAnimation"
                                        interface="bounds">
                <bindParam name="varSet" value="0,0,222.22%,222.22%"/>
            </bind>
        </link>
    </context>
    <context id="menu">
        <port id="pChoro" component="choro"/>
        <port id="pChorinho" component="imgChorinho"/>
        <port id="pRock" component="imgRock"/>
        <port id="pTechno" component="imgTechno"/>
        <port id="pCartoon" component="imgCartoon"/>
        <port id="pNCLua" component="changes"/>
        <port id="pChanges" component="changes" interface="inc"/>
        <media id="changes" src="../script/counter.lua"
                                descriptor="changesDesc">
            <property name="inc"/>
        </media>
        <media id="imgChorinho" src="../media/chorinho.png"
                                descriptor="chorinhoDesc"/>
        <media id="imgRock" src="../media/rock.png"
                                descriptor="rockDesc"/>
        <media id="imgTechno" src="../media/techno.png"
                                descriptor="technoDesc"/>
        <media id="imgCartoon" src="../media/cartoon.png"
                                        descriptor="cartoonDesc"/>
        <media id="choro" src="../media/choro.mp3"
                                        descriptor="audioDesc">
            <property name="soundLevel" value="1"/>
        </media>
        <switch id="musics">
            <bindRule constituent="rock" rule="rRock"/>
            <bindRule constituent="techno" rule="rTechno"/>
            <bindRule constituent="cartoon" rule="rCartoon"/>
            <media id="rock" src="../media/rock.mp3"/>
            <media id="techno" src="../media/techno.mp3"/>
            <media id="cartoon" src="../media/cartoon.mp3"/>
        </switch>
        <link id="lChoro" xconnector="conEx#onSelectionSetStop">
            <bind role="onSelection" component="imgChorinho"/>
            <bind role="set" component="choro" interface="soundLevel">
                <bindParam name="var" value="1"/>
            </bind>
            <bind role="set" component="changes" interface="inc">
                <bindParam name="var" value="1"/>
            </bind>
            <bind role="stop" component="musics"/>
        </link>
        <link id="lOthers"
                        xconnector="conEx#onOrSelectionSetStopStart">
            <bind role="onSelection" component="imgRock"/>
            <bind role="onSelection" component="imgTechno"/>
            <bind role="onSelection" component="imgCartoon"/>
            <bind role="set" component="choro" interface="soundLevel">
                <bindParam name="var" value="0"/>
            </bind>
            <bind role="set" component="changes" interface="inc">
                <bindParam name="var" value="1"/>
            </bind>
            <bind role="stop" component="musics"/>
            <bind role="start" component="musics"/>
        </link>
    </context>
    <link  id="lMusic" xconnector="conEx#onBeginStartDelay">
        <bind role="onBegin" component="animation"/>
        <bind role="start" component="background">
            <bindParam name="delay" value="5s"/>
        </bind>
        <bind role="start" component="menu">
            <bindParam name="delay" value="5s"/>
```

```
            </bind>
        </link>
        <link  id="lDrible" xconnector="conEx#onBeginStart">
            <bind role="onBegin" component="animation"
                                                    interface="segDrible"/>
            <bind role="start" component="drible"/>
        </link>
        <link  id="lPhoto" xconnector="conEx#onBeginStartSetDelay">
            <bind role="onBegin" component="animation"
                                                    interface="segPhoto"/>
            <bind role="start" component="photo"/>
            <bind role="set" component="photo" interface="top">
              <bindParam name="var" value="290"/>
              <bindParam name="delay" value="1s"/>
              <bindParam name="duration" value="3s"/>
            </bind>
        </link>
        <link xconnector="conEx#onEndSetStopDelay">
            <bind role="onEnd" component="animation" interface="segCred"/>
            <bind role="set" component="menu" interface="pChanges">
                  <bindParam name="var" value="FIM"/>
            </bind>
            <bind role="stop" component="menu"/>
            <bind role="stop" component="interactivity"/>
        </link>
        <link  id="lEnd" xconnector="conEx#onEndStop">
            <bind role="onEnd" component="animation"/>
            <bind role="stop" component="background"/>
        </link>
    </body>
</ncl>
```

Listado 3.57 El primer *João* con objeto NCLua.

La Figura 3.21 ilustra la utilización de un objeto NCLua en la implementación de referencia del middleware Ginga-NCL, donde se puede observar el mensaje que indica el número de veces que se cambio el ritmo.

Figura 3.21 Escenas de la aplicación El primer *João* con objeto NCLua.

Bibliografía

ABNT NBR 15606-2 (2011). Associação Brasileira de Normas Técnicas, "Televisão digital terrestre — codificação de dados e especificações de transmissão para radiodifusão digital, Parte 2: Ginga-NCL para receptores fixos e móveis — Linguagem de aplicação XML para codificação de aplicações", Sistema Brasileiro de TV Digital Terrestre, *NBR 15606-2*.

ITU-T H.761 (2011). "Nested Context Language (NCL) and Ginga-NCL for IPTV Services." Recommendation H.761, Genebra,.

W3C REC-SMIL2-20051213 (2008). World Wide Web Consortium, "Synchronized Multimedia Integration Language — SMIL 2.1 Specification", *W3C Recommendation SMIL2-20051213*.

W3C REC-xml-20060816 (2006). World Wide Web Consortium, "Extensible Markup Language (XML) 1.0", W3C Recommendation *xml-20060816*.

W3C REC-xml-names-20060816 (2006). World Wide Web Consortium, "Namespaces in XML (2.ª ed.)", W3C Recommendation *xml-names 20060816*.

Soares, L.F.S. e Rodrigues, R.F. (2005). "Nested Context Model 3.0 Part 1 — NCM Core." Monografias em Ciência da Computação do Departamento de Informática, PUC-Rio, N.º 18/05. Rio de Janeiro. Maio. ISSN 0103-9741.