

Programación en Ginga-NCL

- Definición de la presentación
- Inserción de los elementos
- Organización del documento
- Sincronización de los elementos
- Definición de alternativas

Programación en Ginga-NCL

Sincronización de los elementos

- Conectores
- Enlaces
 - definir cómo los elementos se relacionan en la presentación
 - recepción de la interacción de los usuarios

Conectores

- Establecen relaciones genéricas utilizadas por los elementos de un documento NCL
 - no especifica los participantes de una relación
 - Ejemplo, la relación enseña a, define a alguien que enseña y alguien que aprende pero no indica quien enseña y quien aprende.

Conectores

- En NCM y en NCL, el sincronismo
 - no está hecho por timestamps
 - si por mecanismos de causalidad y restricción
- Establece las funciones (roles) que los nodos de origen y de destino ejercen en los enlaces.
- elemento <connectorBase>

Conectores

- Una base de conectores contiene los elementos hijos:
 - <causalConnector>: define un conector propiamente.
 - <importBase>: permite importar una base de conectores de algún otro archivo.

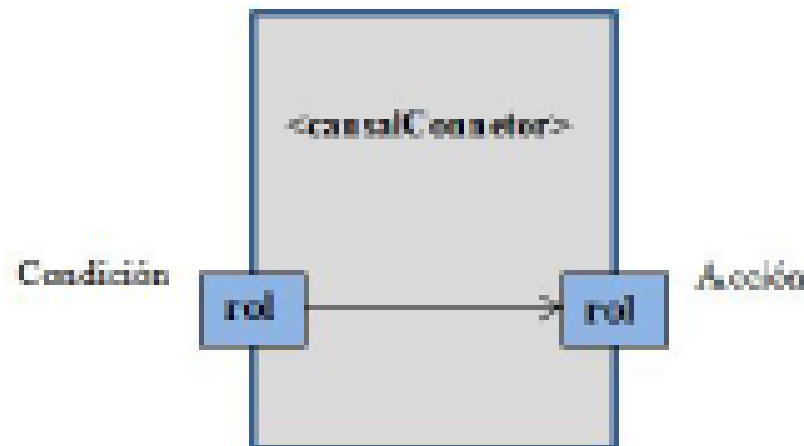
Conectores

- Ejemplo

```
<connectorBase id="menusConectores">  
  <importBase .../>  
  <importBase .../>  
  <causalConnector id="onBeginStar">  
    ...  
  </causalConnector>  
  <causalConnector id=" ">  
    ... </causalConnector id=" ">  
    ...  
  <causalConnector id=" ">  
</connectorBase>
```

Conectores

- En NCL 3.0, existe solo un tipo de conector:
 - conector causal (causalConnector) define:
 - condiciones (condition) bajo las cuales el enlace <link> puede ser activado
 - acciones (action) que serán realizadas

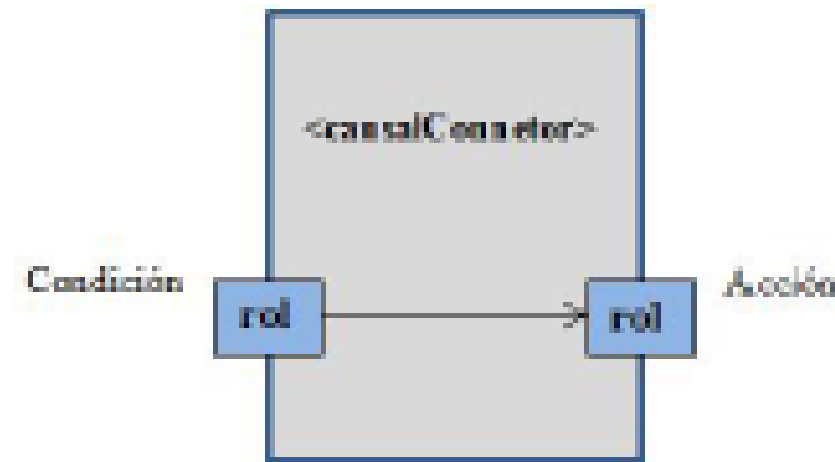


Conectores

- elementos hijo de un <causalConnector > son:
 - <connectorParam>: parámetros cuyos valores deberían ser establecidos por los enlaces que utilizan un conector.
 - <simpleCondition> y <compoundCondition>: condiciones simples o compuestas de activación de un enlace que utiliza un conector
 - <simpleAction> y <compoundAction>: acciones simples o compuestas que se realizarán cuando un enlace que utiliza un conector sea activado.

Conectores: Condiciones simples

- elemento `<simpleCondition>` establece
 - condición a cumplirse → conector sea activado
 - atributo `role` el nombre del papel de la condición.
- NCL tiene un conjunto de nombres reservados para condiciones

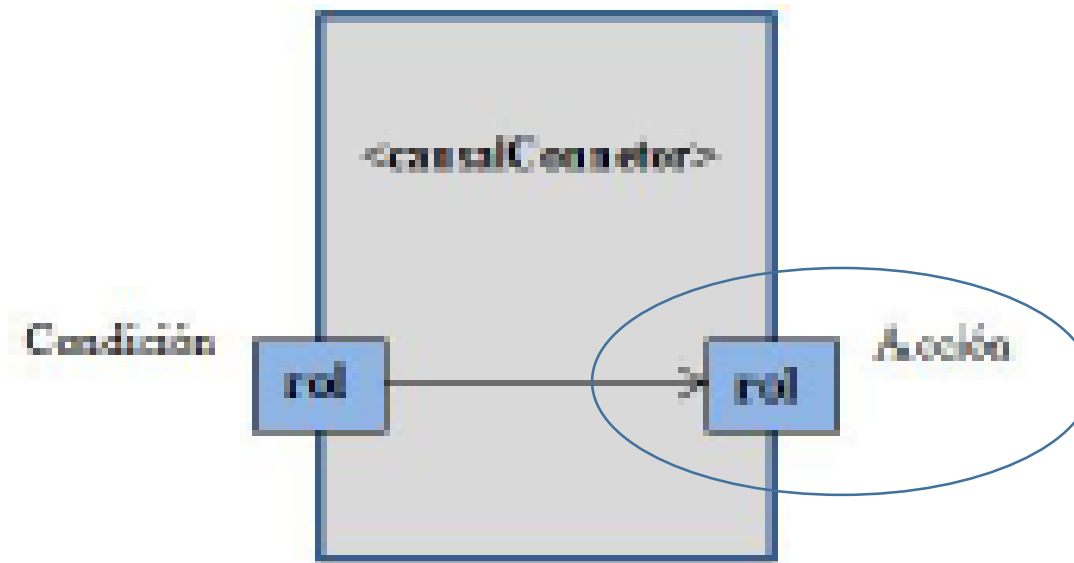


Conectores: Condiciones simples

- onBegin: Se activa cuando la presentación de los elementos vinculados a esta función son iniciados.
- onEnd: ... son terminados.
- onAbort: ...son abortados.
- onPause: ... se detienen.
- onResume: ...retornan después de una pausa

Conectores: Acciones simples

- elemento `<simpleAction>` establece
 - acción ejecutarse → conector sea activado
 - atributo role el nombre de la función de acción



Conectores: Acciones simples

Atributos

- max: el número máximo de elementos para poder utilizar este documento.
 - valor = "unbounded" número máximo ilimitado.
 - Si valor se necesita otro atributo qualifier
- qualifier: establece si la acción será ejecutada en paralelo o secuencialmente
 - valores = "par" y "seq"

Conectores: Acciones simples

- Nombres reservados para funciones de acciones
 - start: inicia la presentación del elemento vinculado a esta función.
 - stop: finaliza
 - abort: cancela
 - pause: pausa
 - resume: retoma
 - set: establece un valor o una propiedad de un elemento asociado a esta función.
 - atributo value

Conectores: Condiciones y Acciones simples

Ejemplo

- "onBegin", condición esperada al inicio de la presentación de un elemento,
- "start", indica que la presentación de un elemento será iniciada.

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

Programación en Ginga-NCL

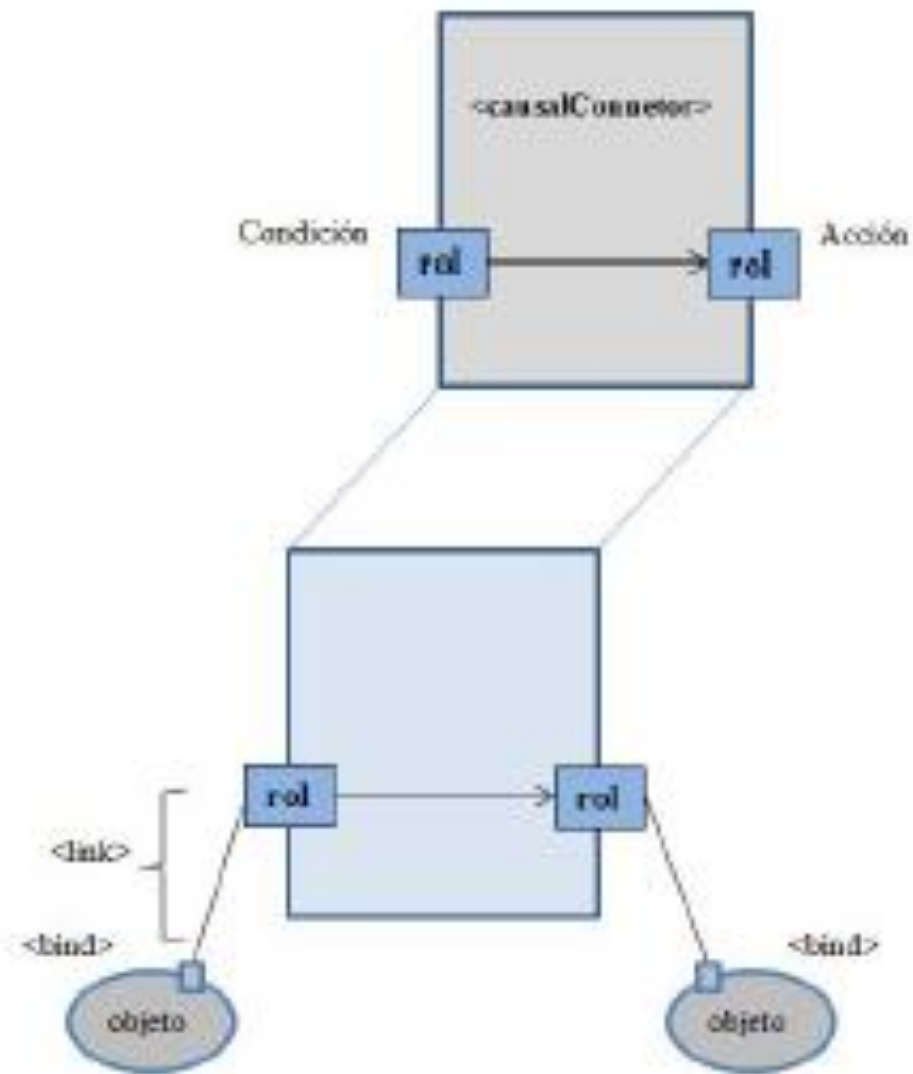
Sincronización de los elementos

- Conectores
- Enlaces
 - definir cómo los elementos se relacionan en la presentación
 - recepción de la interacción de los usuarios

Enlaces

- Un enlace (link) se utiliza para identificar los elementos que participan en una relación.
- Ejemplo "enseña a" debería identificar los elementos "maestro" y "estudiante".
- La correspondencia completa sería "el maestro enseña al alumno".

Enlaces



Enlaces : Atributos

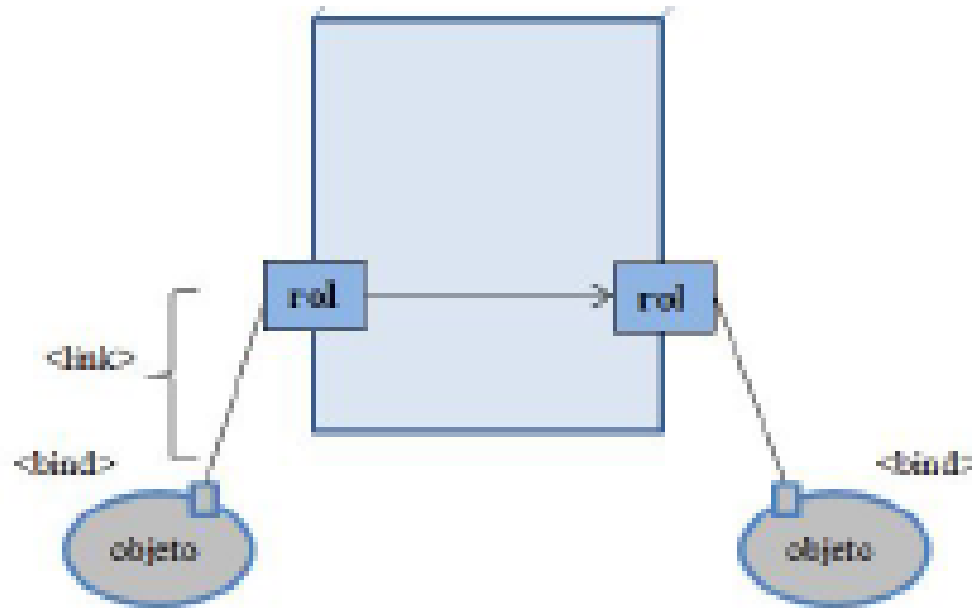
- id: identificador
- xconnector: identificador de conector

Elementos contenidos en un enlace

- Bind: indica un componente
 - nodo multimedia o de contexto involucrado en el enlace
 - papel (role) en el mismo, conforme la semántica del conector
 - [el punto de interfaz (interface) del nodo]
- ```
<bind role="onBegin" component="video" interface="areaimg"/>
```

# Enlaces

- Para la creación de las conexiones entre los elementos y los documentos, un enlace crea elementos secundarios <bind>



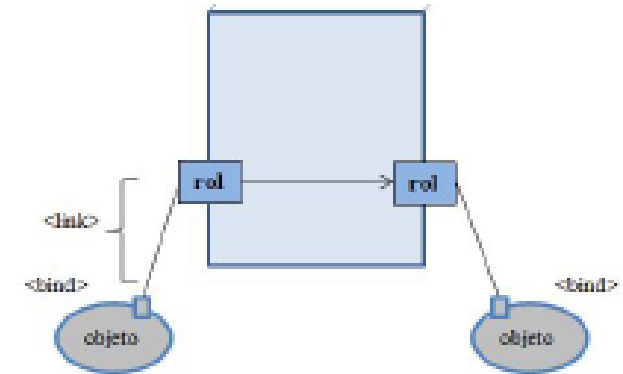
# Enlaces y Conectores

- Conector

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


- Enlace

```
<link xconnector="onBeginStart">
 <bind role="onBegin" component="video" />
 <bind role="start" component="imagen" />
</link>
```



# Conectores: Condiciones simples

- onSelection: se activa cuando
  - pulsa una tecla que se especifique  
→ key
    - "0" al "9", "A" hasta la "Z", "\*", "#",  
etc.
  - tecla ENTER y el elemento esta con  
el foco.

Valores para la propiedad key:	Correspondencia con botones del control remoto:	
RED	F1	
GREEN	F2	
YELLOW	F3	
BLUE	F4	
MENU	F5	
INFO	F6	
ENTER		
CURSOR_LEFT		
CURSOR_UP		
CURSOR_RIGHT		
CURSOR_DOWN		

# Ejemplo

```
<casualConnector id="onKeySelectionStart">
```

```
<simpleCondition role="onSelection" key="GREEN">
```

```
<simpleAction role="start">
```

```
</casualConnector>
```

```
<link xconnector="onKeySelectionStart" id="verde">
```

```
 <bind role="onSelection" component="imblue"/>
```

```
 <bind role="start" component="img"/>
```

```
</link>
```

# Conectores

- elementos hijo de un <causalConnector > son:
  - <connectorParam>: parámetros cuyos valores deberían ser establecidos por los enlaces que utilizan un conector.
  - <simpleCondition> y <compoundCondition>: condiciones simples o compuestas de activación de un enlace que utiliza un conector
  - <simpleAction> y <compoundAction>: acciones simples o compuestas que se realizaran cuando un enlace que utiliza un conector sea activado.

# Conectores: parámetros

- Se utiliza para que el valor sea validado o establecido en el momento de uso del conector

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

- NOTA: el parámetro solo define su nombre, dejando su valor para el momento de su uso.



# Conectores: parámetros

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

# Conectores: parámetros

```
<link xconnector="onKeySelectionStart" id="azul">
 <linkParam name="keyCode" value="BLUE"/>
 <bind role="onSelection" component="imblue"/>
 <bind role="start" component="img"/>
</link>
```

```
<link xconnector="onKeySelectionStart" id="azul">
 <bind role="onSelection" component="imblue">
 <bindParam name="keyCode" value="BLUE"/>
 </bind>
 <bind role="start" component="img"/>
</link>
```

# Ejemplo

```
--
<head>
.....
<connectorBase>
.....
 <causalConnector id="onKeySlectionStop">
 <connectorParam name="keyCode"/>
 <simpleCondition role="onSelection" key="$keyCode" />
 <simpleAction role="stop" />
 </causalConnector>
 </connectorBase>
</head>
..
```

# Ejemplo

```
<body>
 <port id="InVideo" component="videoIntro"/>

 <media id="boton" type="image/jpeg" src="media/boton.jpg"
 descriptor="desImagen" >
 <property name="transparency" value="50 %"/>
 </media>

 <link xconnector="onKeySlectionStop">
 <bind role="onSelection" component="boton" >
 <bindParam name="keyCode" value="GREEN"/>
 </bind>
 <bind role="stop" component="videoIntro" />
 </link>
</body>
```

# Varias acciones simpleCondition

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

```
<link xconnector="onEndStopN">
 <bind role="onEnd" component="videoNoticias" />
 <bind role="stop" component="imgInteratividade" />
 <bind role="stop" component="imgMenu" />
 <bind role="stop" component="txtGRU"/>
</link>
```

# Conectores: Condiciones compuestas

- etiqueta `<compoundCondition>`
- Posee dos o más condiciones simples como hijas.
- Se debe declarar un atributo *operator*
  - valores "and" y "or"
  - si todas o por lo menos una condición debe ser satisfecha para que el conector sea activado

# Conectores: Acciones compuestas

- Definida por elemento <compoundAction>
- Posee otras acciones simples como hijas
- Se debe definir un atributo operator
  - valores = "par" o "seq"
  - indicando si las funciones deberán ser ejecutadas en paralelo o secuencialmente

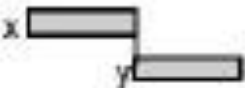




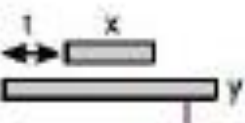

# Varias acciones compoundCondition

```
<causalConnector id="onEndStartNStopN">
 <simpleCondition role="onEnd"/>
 <compoundAction operator="seq">
 <simpleAction role="start" max="unbounded" qualifier="par"/>
 <simpleAction role="stop" max="unbounded" qualifier="par"/>
 </compoundAction>
</causalConnector>
```

```
<link xconnector="onEndStartNStopN">
 <bind role="onEnd" component="videoNoticias" />
 <bind role="start" component="txt1" />
 <bind role="stop" component="img1" />
 <bind role="stop" component="img2" />
</link>
```



# Ejemplos de conectores

Relacion de iniciación	Ilustración	Conector Hipermedia
x inicia y y inicia cuando termina x		onEndStart
x y y inician simultaneamente		onBeginStart
x y y terminan simultaneamente		onEndStop
x termina y y inicia despues de un tiempo t		onEndStartDelay
x inicia y y inicia despues de un tiempo t		onBeginStartDelay
x es contenida durante la presentacion de y y su inicio es defasado por un tiempo t		onBeginStartDelay onEndStopDelay
x y y inician y terminan igualmente		onBeginStart onEndStop

Importando bases de  
archivos externos?

# Ejemplo

```
<connectorBase>
 <importBase documentURI="conector_menu.ncl"
 alias="conector"/>
</connectorBase>

<link xconnector="conector#OnBeginStart" id="limg">
 <bind role="onBegin" component="video" />
 <bind role="start" component="img1" />
</link>
```