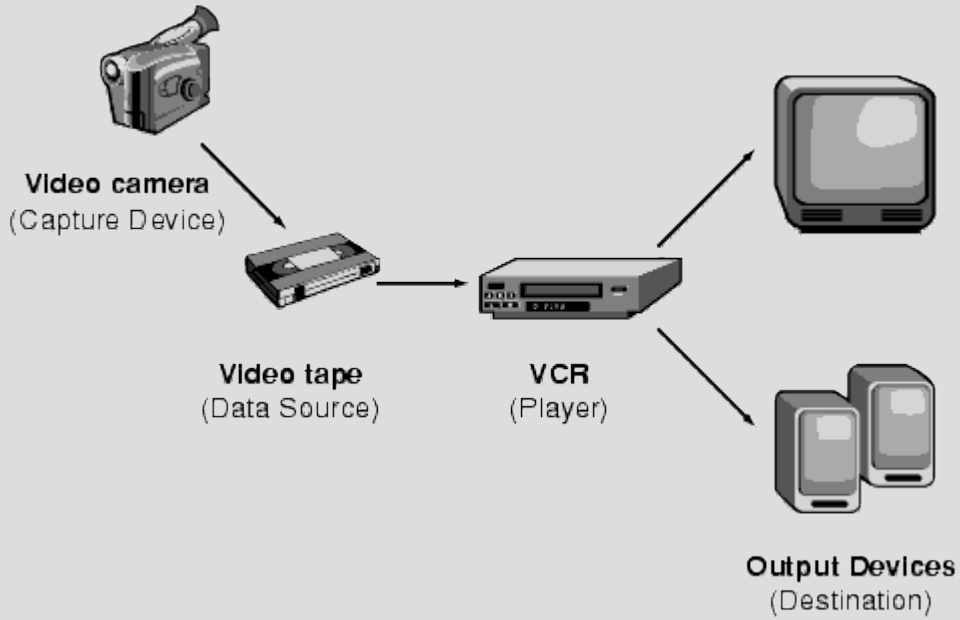
JMF: Java Media Framework

F. Javier Ríos Pérez

Arquitectura básica

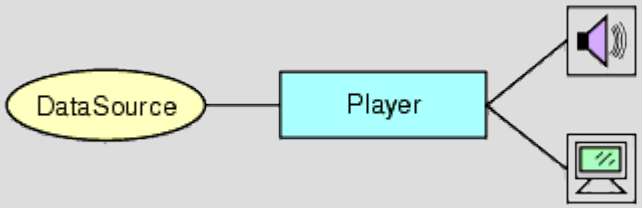


DataSource

- Los datos multimedia pueden proceder de:
  - Archivos locales o remotos.
  - Video y audio en tiempo real o bajo demanda.
- Una fuente de datos se modela mediante un objeto DataSource.
- Podemos crear un DataSource a partir de una URL o de un objeto MediaLocator.
- Soporta todo tipo de contenido multimedia.

Player

- Actúa como reproductor de datos multimedia.
- Asegura que lleguen adecuadamente al dispositivo de salida.
- No hace falta un player específico para cada tipo de datos.



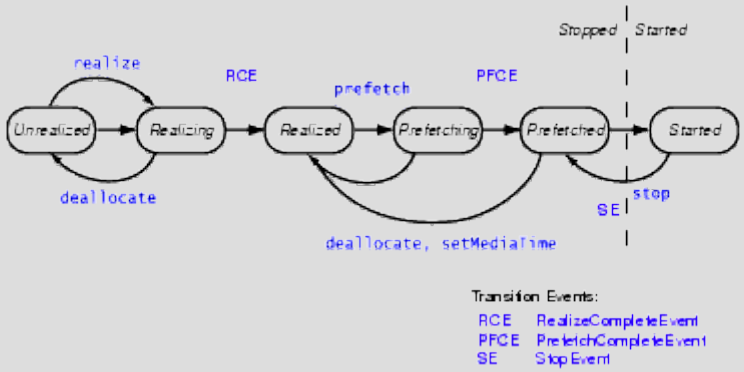
Player: Creación

- Para crear un Player se usa el método createPlayer() de la clase Manager.
- El flujo de datos se le pasará como argumento a dicho método pudiendo ser: DataSource, MediaLocator o URL.

```
URL url = new URL(“http://...”);
Player p = Manager.createPlayer(url);
```

Player: Estados

- El diagrama de estados de un player es el siguiente:



- Aunque hay métodos para ir pasando de un estado a otro, normalmente bastará con llamar a start para pasar directamente a started.

Player: Métodos

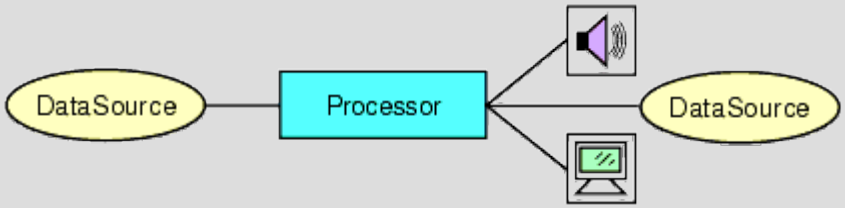
- Los métodos de un Player son:
  - start(): Inicia la reproducción. (Started)
  - stop(): Desciende la reproducción. (Prefetched)
  - realize(): Va al estado Realized.
  - Prefetch(): Va al estado Prefetched.
  - Deallocate(): Cierra el player y libera recursos.
  - setRate(float rate): Establece la velocidad de reproducción.
  - setMediaTime(Time time): Indica a partir de donde continuar la reproducción.

Player: GUI

- Para facilitar el uso de un Player se han definido dos componentes gráficos:
  - Componente visual: muestra la pista de vídeo del contenido multimedia si la hubiera.
    - Se obtiene con el método getVisualComponent() del Player.
  - Componente de control: permite al usuario controlar la reproducción (iniciar, detener, volumen...)
    - Se obtiene llamando al método getControlPanelComponent() del Player.

Processor

- Es una subinterfaz de Player que aporta dos nuevas características:
  - Puede volcar los datos procesados en un DataSource, en lugar de en los dispositivos de salida.
  - Puede cambiar el formato de los datos.



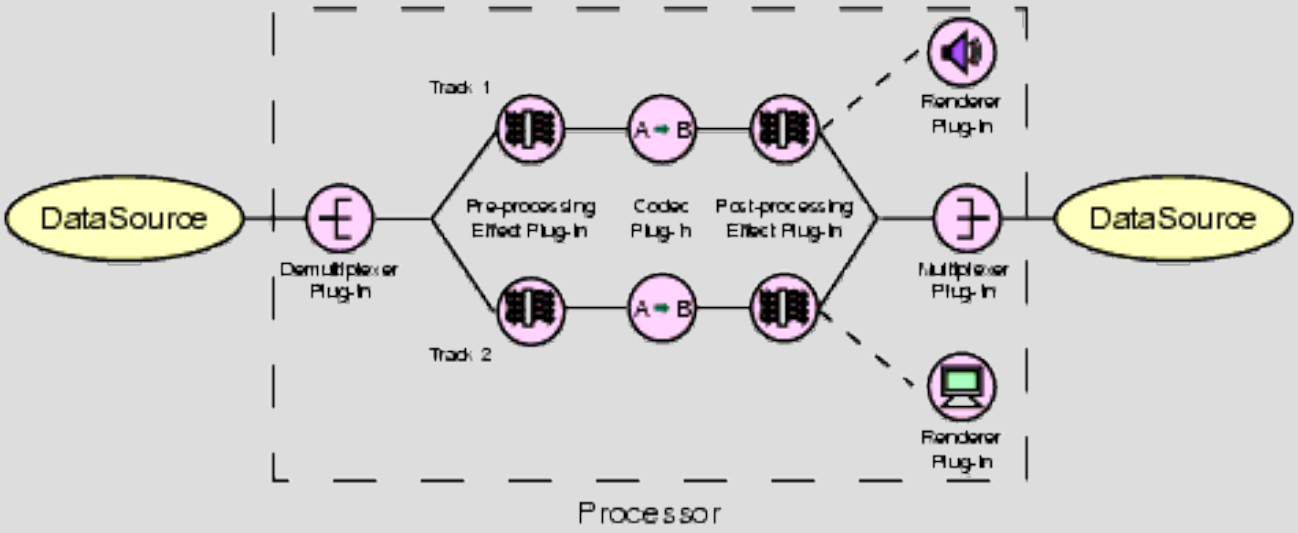
Processor: Creación

- Para crear un Processor se usa el método createProcessor() de la clase Manager.
- El flujo de datos se le pasará como argumento a dicho método pudiendo ser: DataSource, MediaLocator o URL.

```
URL url = new URL(“http://...”);
Processor p = Manager.createProcessor(url);
```

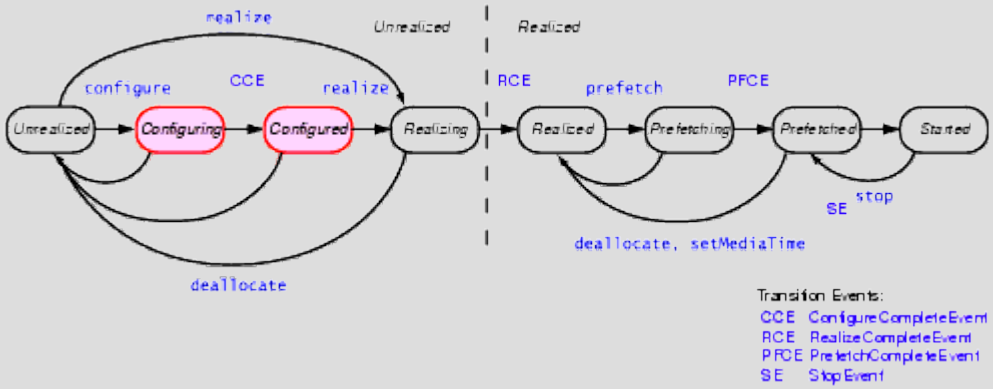
Processor: Fases

- El proceso de los datos multimedia se divide en las siguientes fases:



Processor: Estados

- Tiene dos estados más que el Player



DataSink

- Es una interfaz de javax.media
- Envía los datos a un destino distinto delos dispositivos de salida:
  - Ficheros
  - Flujo de salida RTP

Format

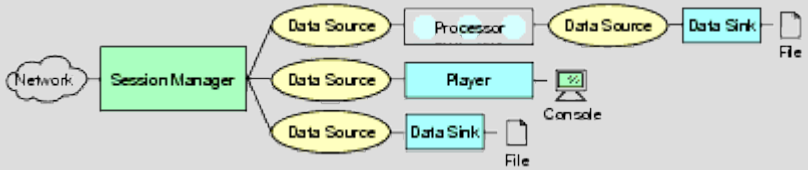
- Es una clase de javax.media
- Modela el formato de los datos multimedia.
- Tiene dos subclases:
  - AudioFormat: Encapsula información sobre el formato de datos de audio:
    - Frecuencia de mustreo.
    - Bits/Muestra.
  - VideoFormat: Encapsula información sobre el formato de los datos de vídeo.

RTP

- El API de JMF RTP provee los mecanismos necesarios para la trasmisión de audio y video a través de la red.
- Hace transparente al programador las complicación de la trasmisión por la red, ya que para éste será prácticamente igual que guardar el flujo en un fichero local.

RTP: Recepción

- Para la recepción de un flujo de audio/vídeo usando RTP lo único que debemos conocer es la URL de dicho flujo:
  - rtp://address:port[:ssrc]/content-type/[ttl]
- A partir de dicha URL crearemos un MediaLocator y con él obtendremos el Player/Processor.



RTP: Trasmisión

- Para enviar una trama de Audio/Vídeo debemos:
  - Crear, inicializar y arrancar una SessionManager para la sesión.
  - Crear un Processor usando el DataSource a transmitir.
  - Configurar el Processor para devolver un formato RTP.
  - Coger el DataSource de salida del Processor.
  - Llamar a createSendStream en el SessionManager y pasarle el DataSource.