

# User Stories

# Endymion - Member of the IPS team

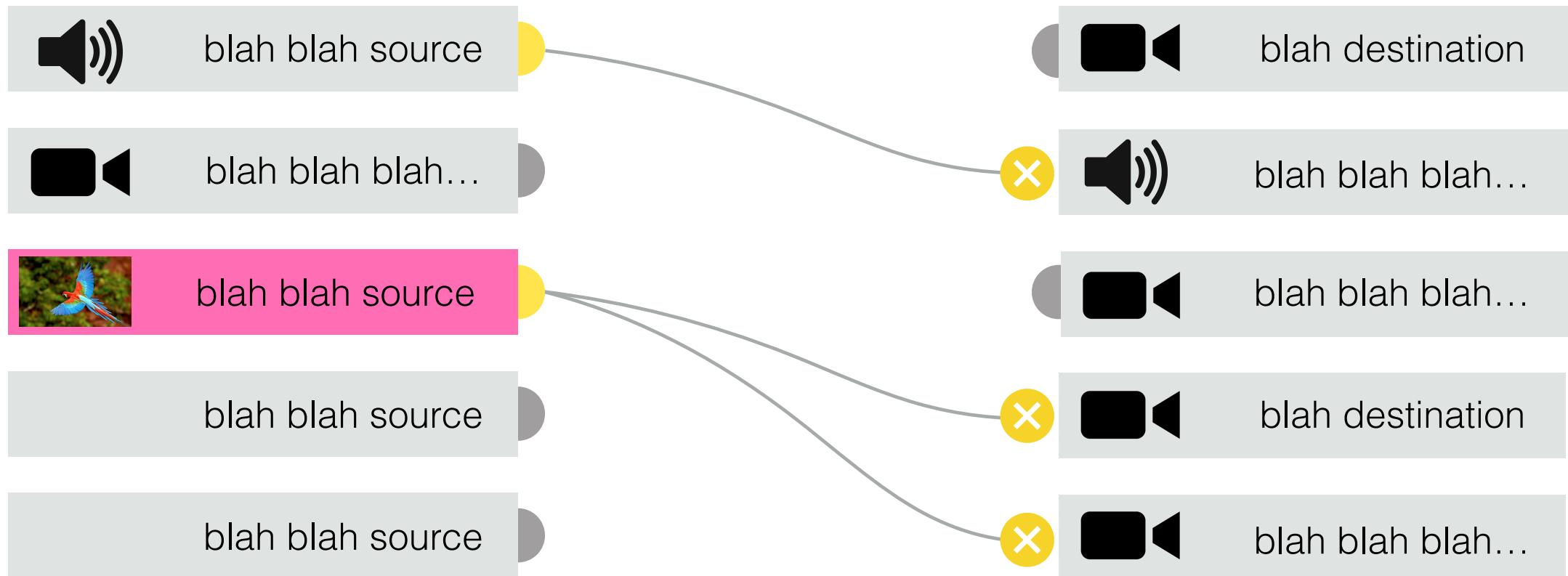
## CREATE A NEW ROUTE

- open web router in browser
- select/ find sources and destinations that they are interested in
- having selected these, the UI displays these to them
- identify & select source that they would like to route from.
- UI displays valid potential destinations for route/s
- select a destination
- the UI displays to the user that a route is being created
- the UI displays to the user that a route has been created
- This can happen as many times as desired (including from the same source)

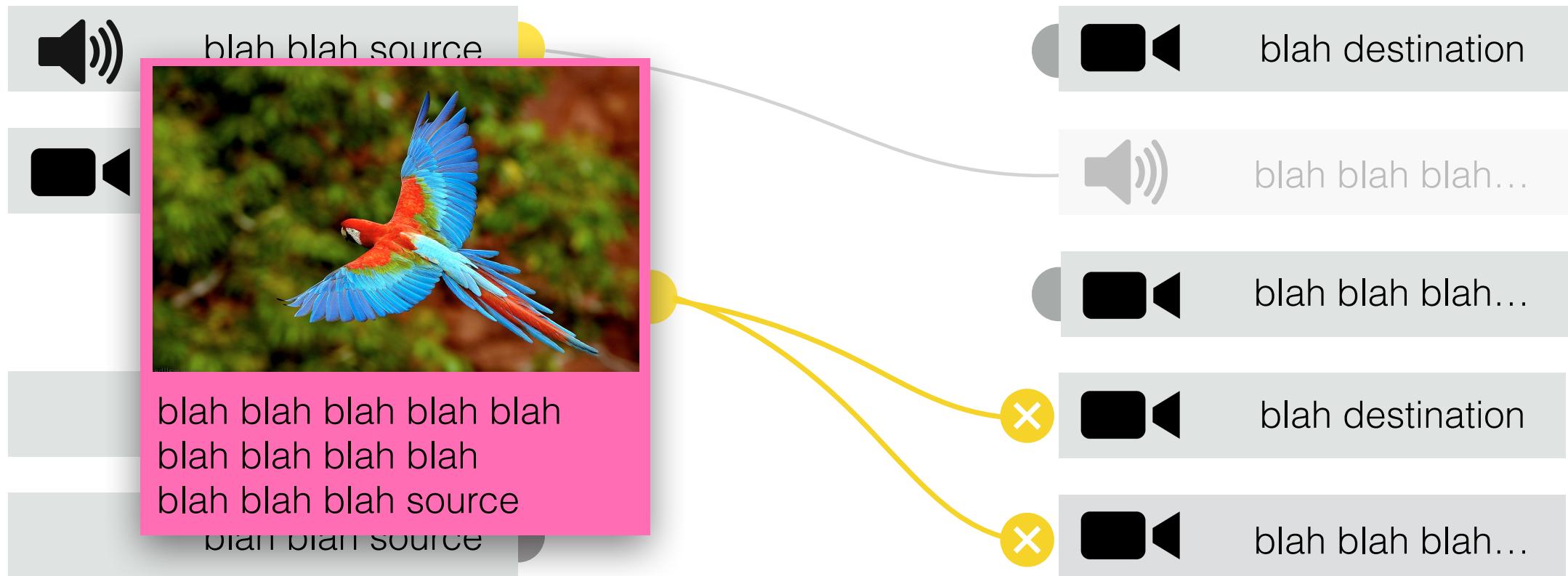
## DELETE A ROUTE

- open web router in browser
- select/ find sources and destinations that they are interested in
- having selected these, the UI displays these to them
- identify & select the route that they would like to destroy using UI 'destroy' element.
- the UI displays to the user that a route is being destroyed
- the UI displays to the user that a route has been destroyed
- This can happen as many times as desired (including from the same source)

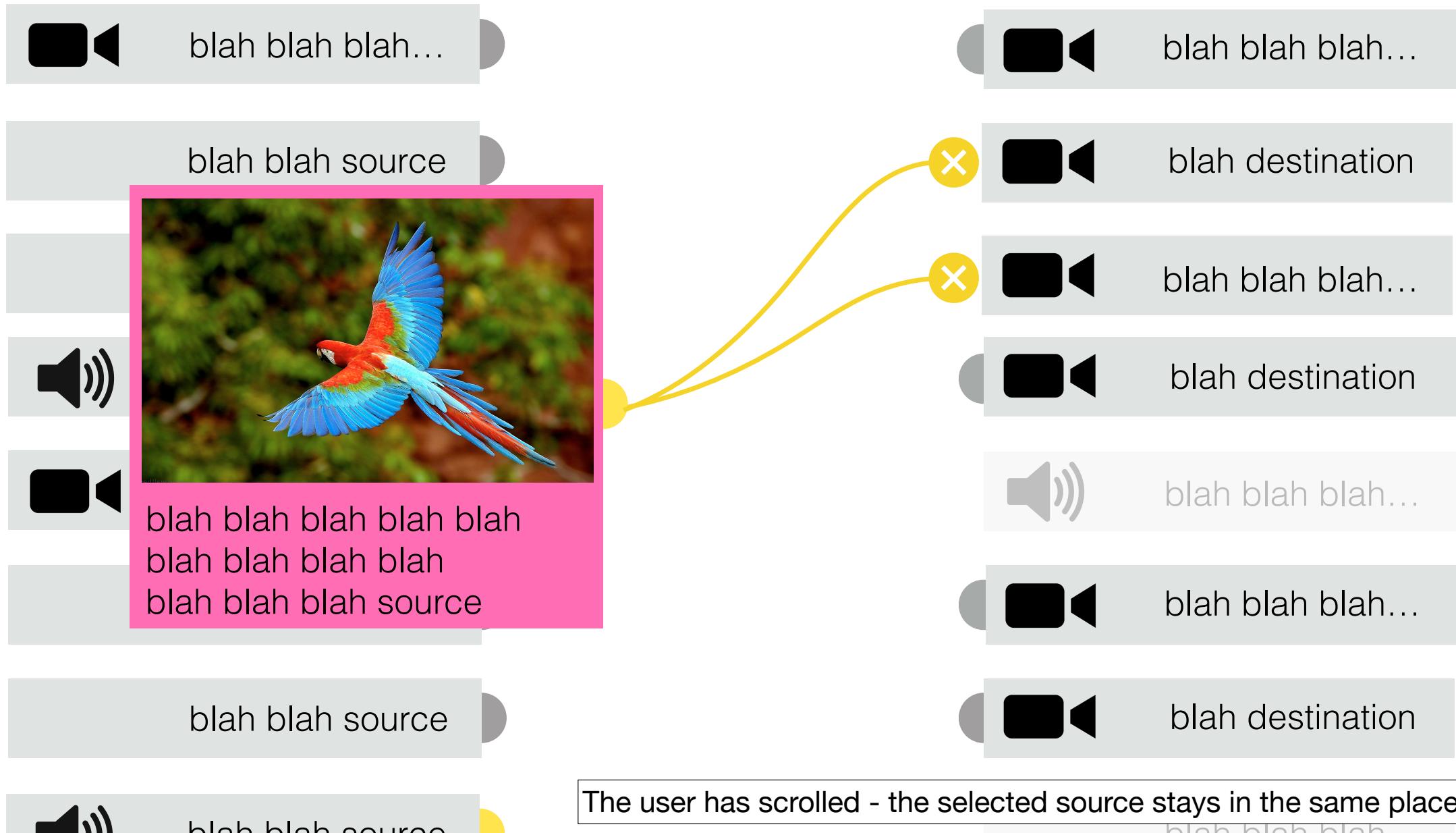
# Creating a Route

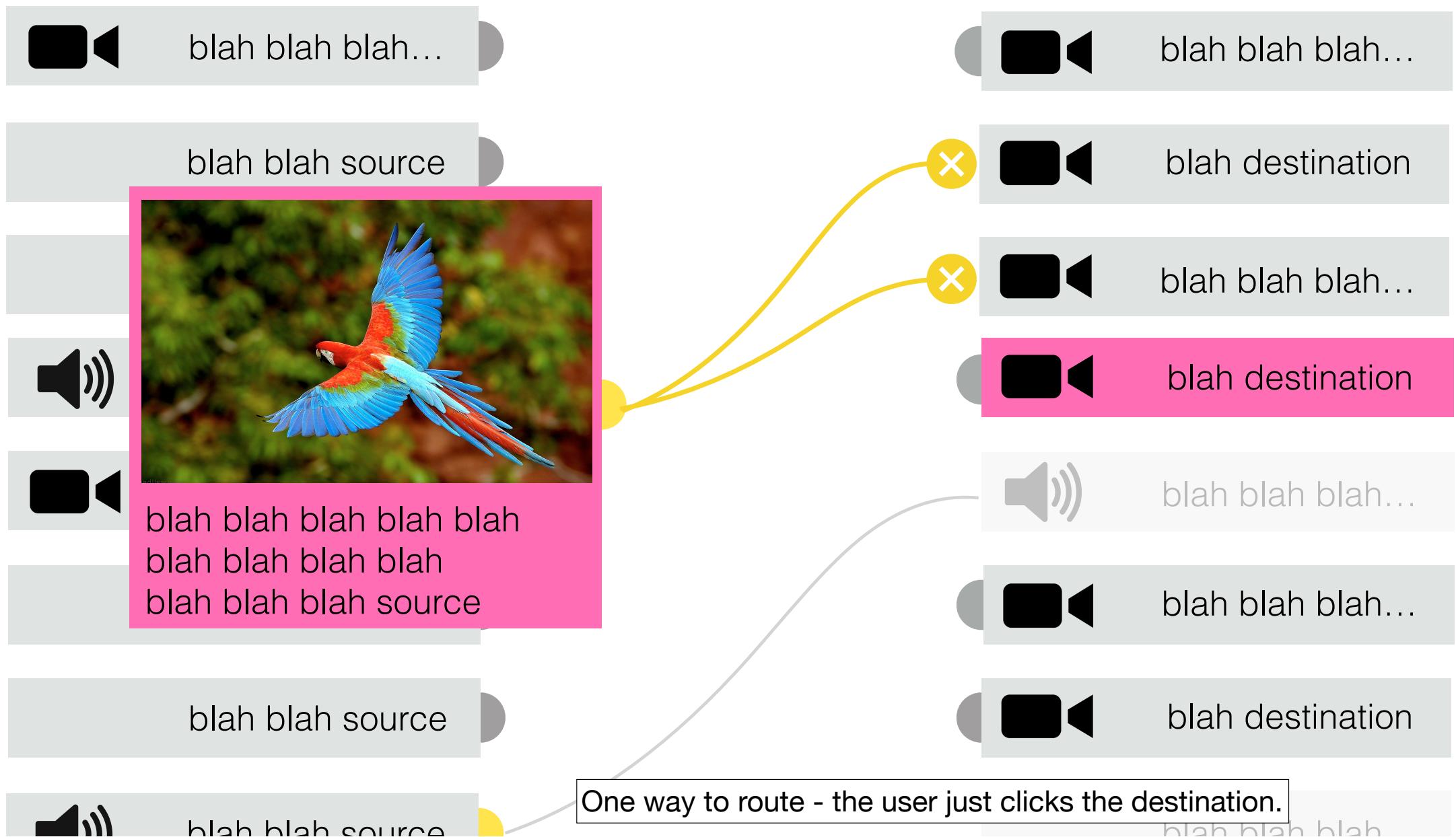


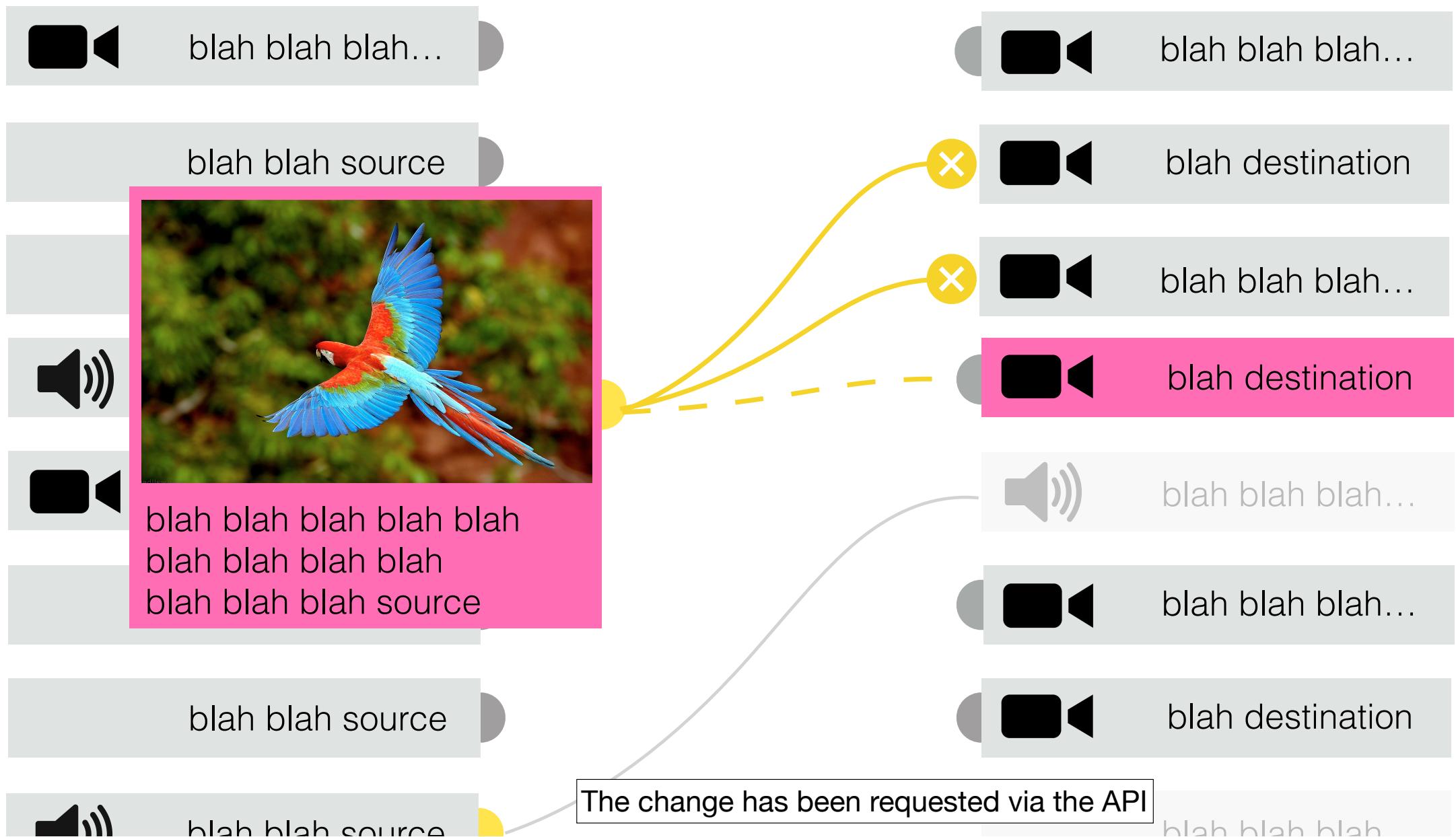
The user selects a source by clicking or tapping it.

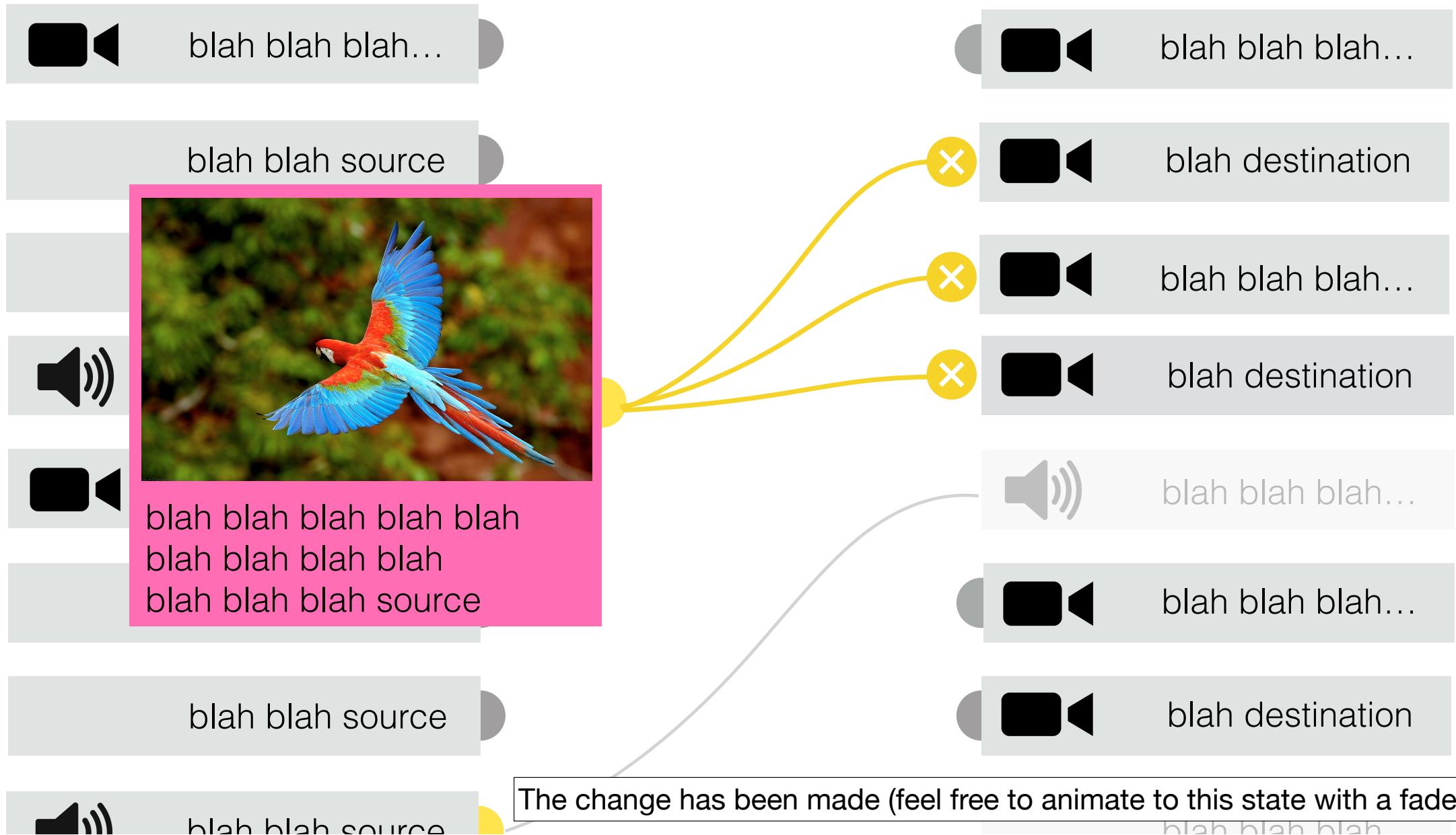


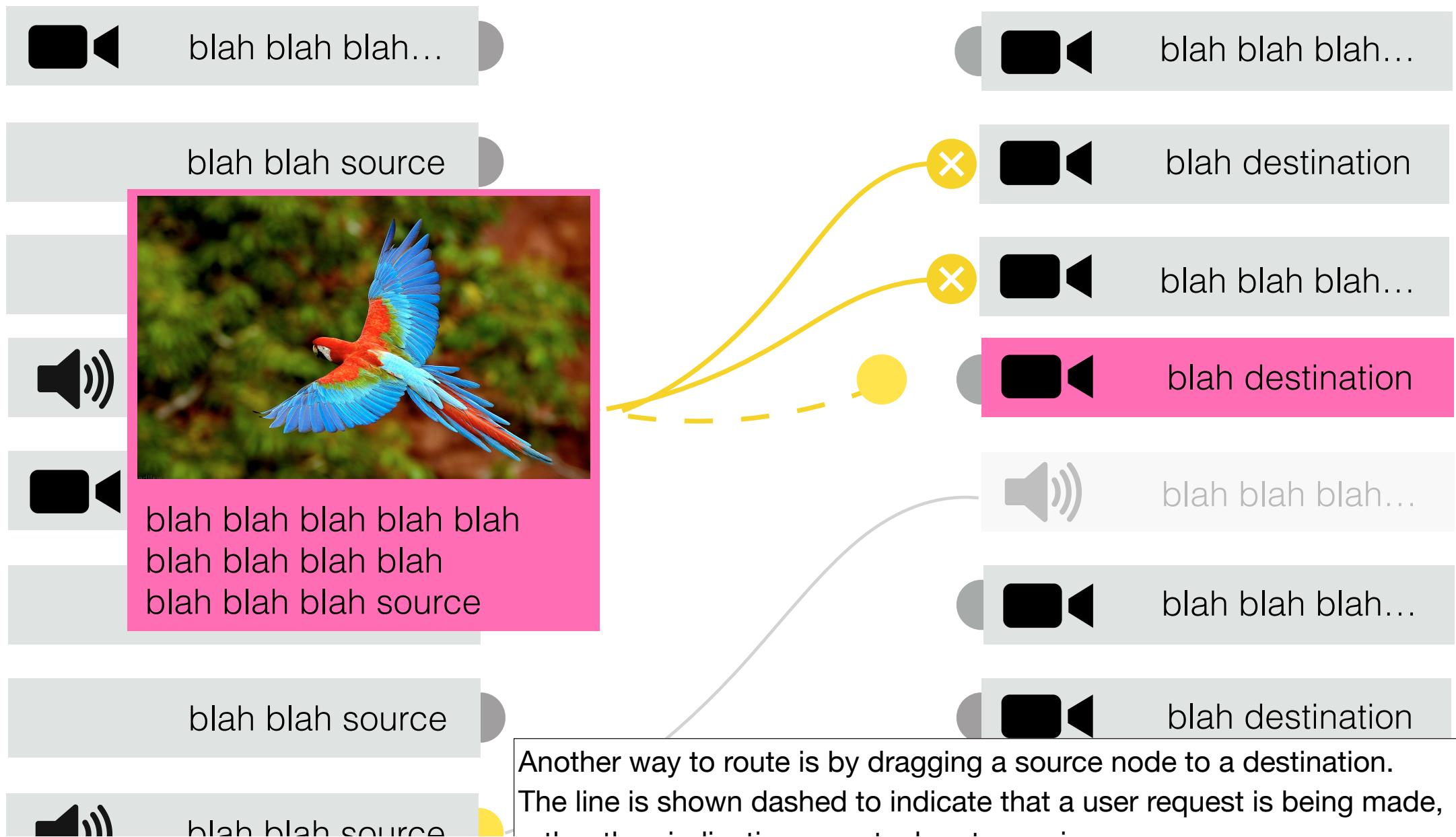
The source expands, is indented and gains a drop shadow. It now remains stationary when the list scrolls. Its routes are indicated by a thick line the same colour as an active node, rendered on top of all other routes. (Selecting another source or clicking “on the background” causes the source to return to its place in the list.) Destinations that are incompatible with that source lose their nodes and fade to 25% opacity over a time period of 0.25s. Routes from other senders fade to 50% opacity over a time period of 0.25s

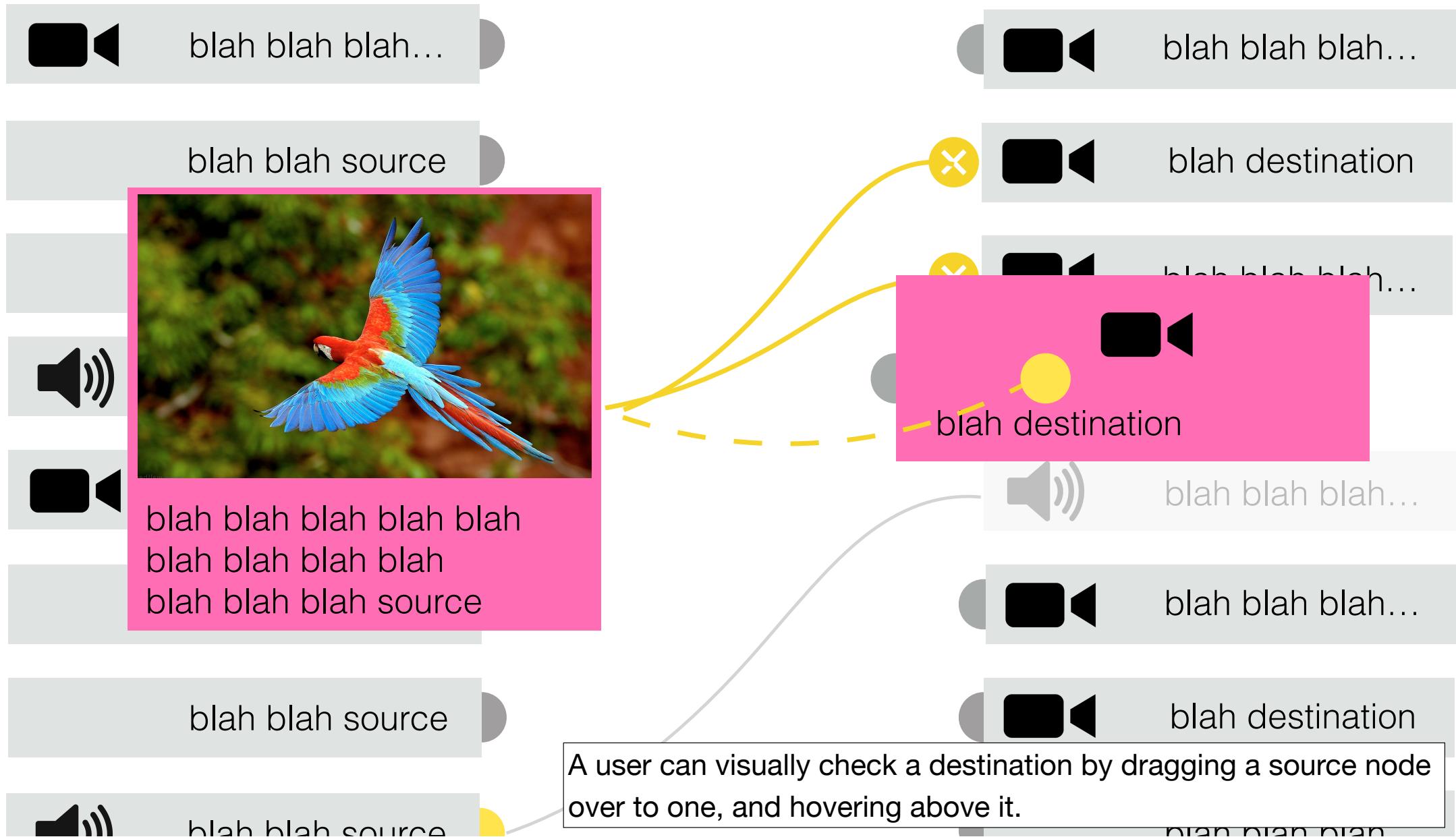


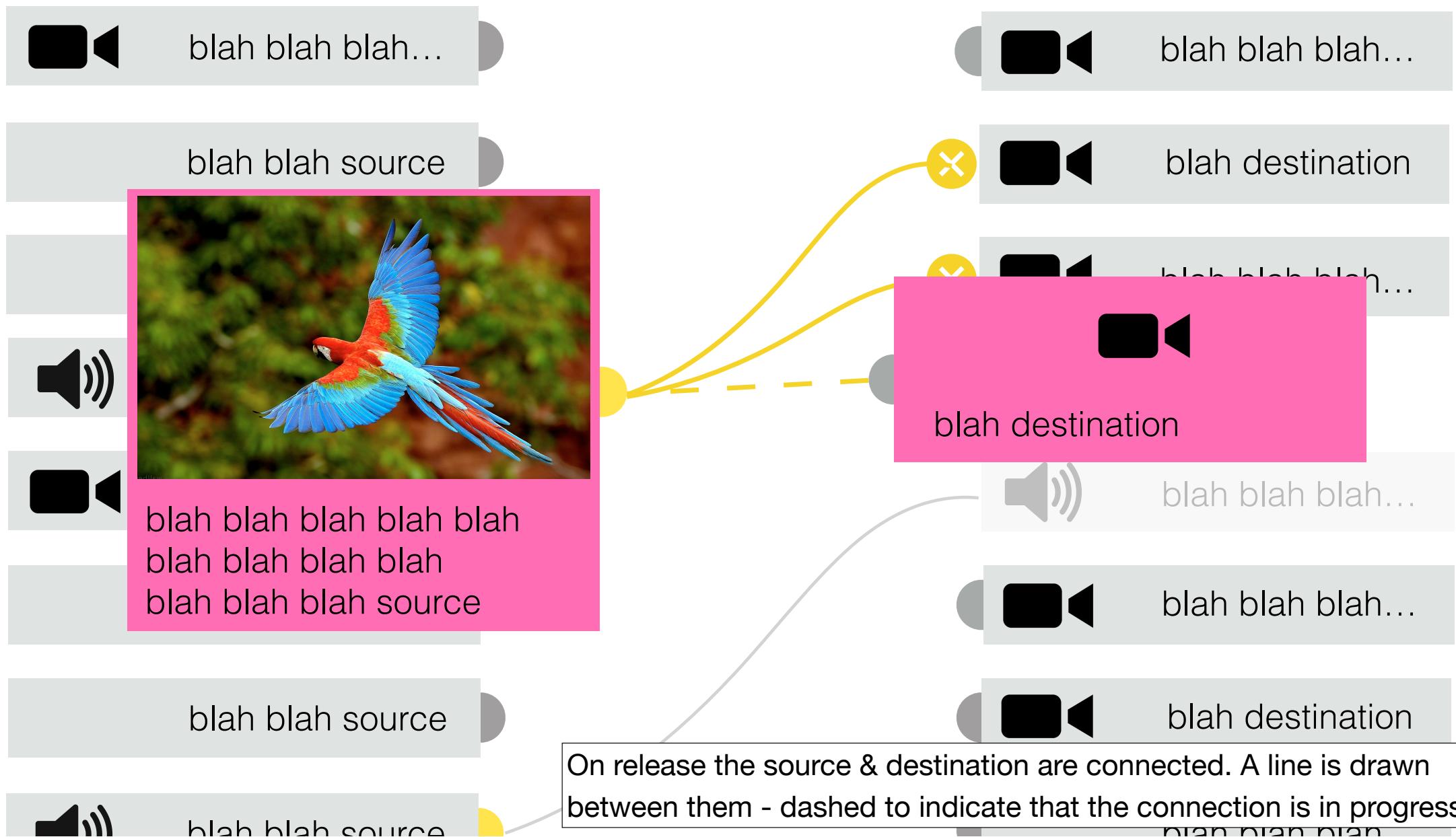


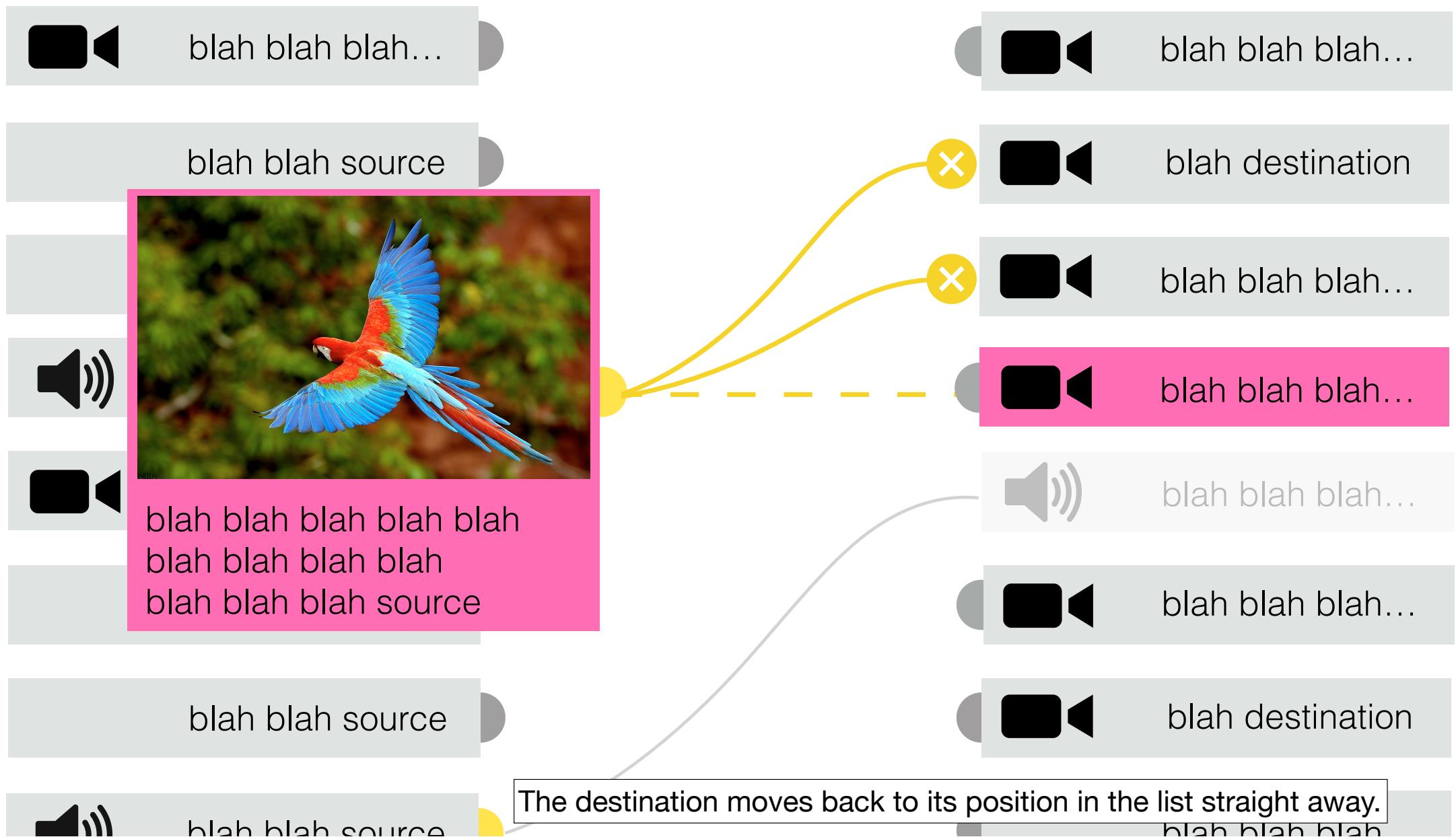


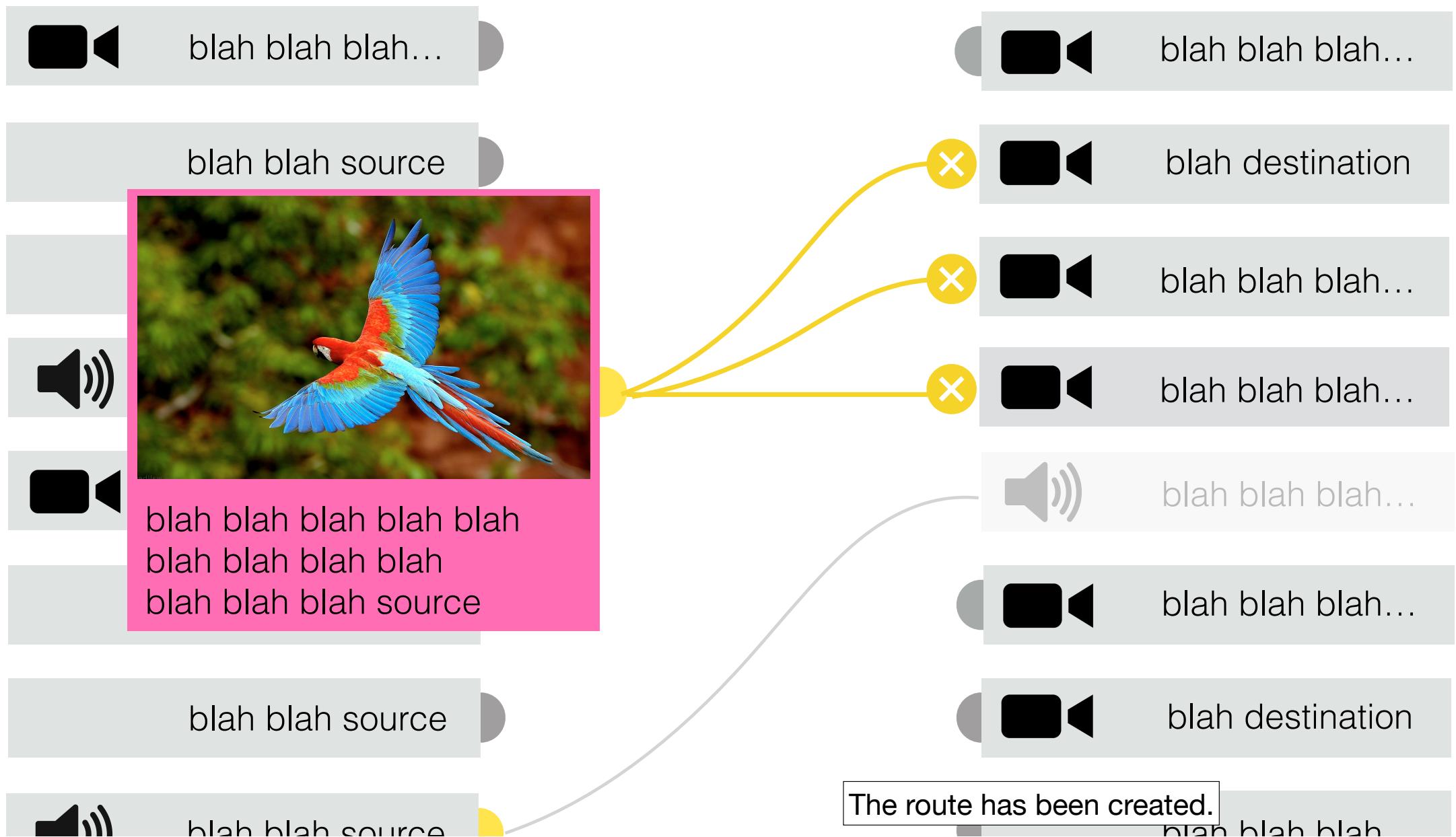


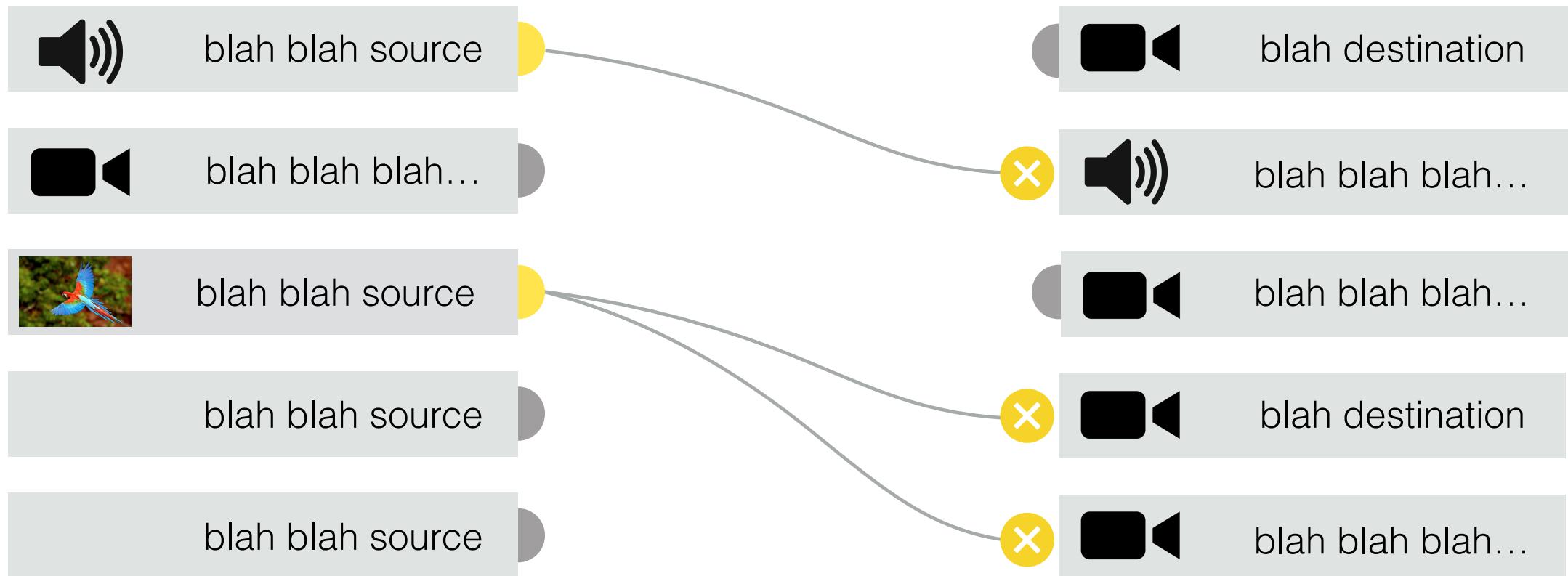






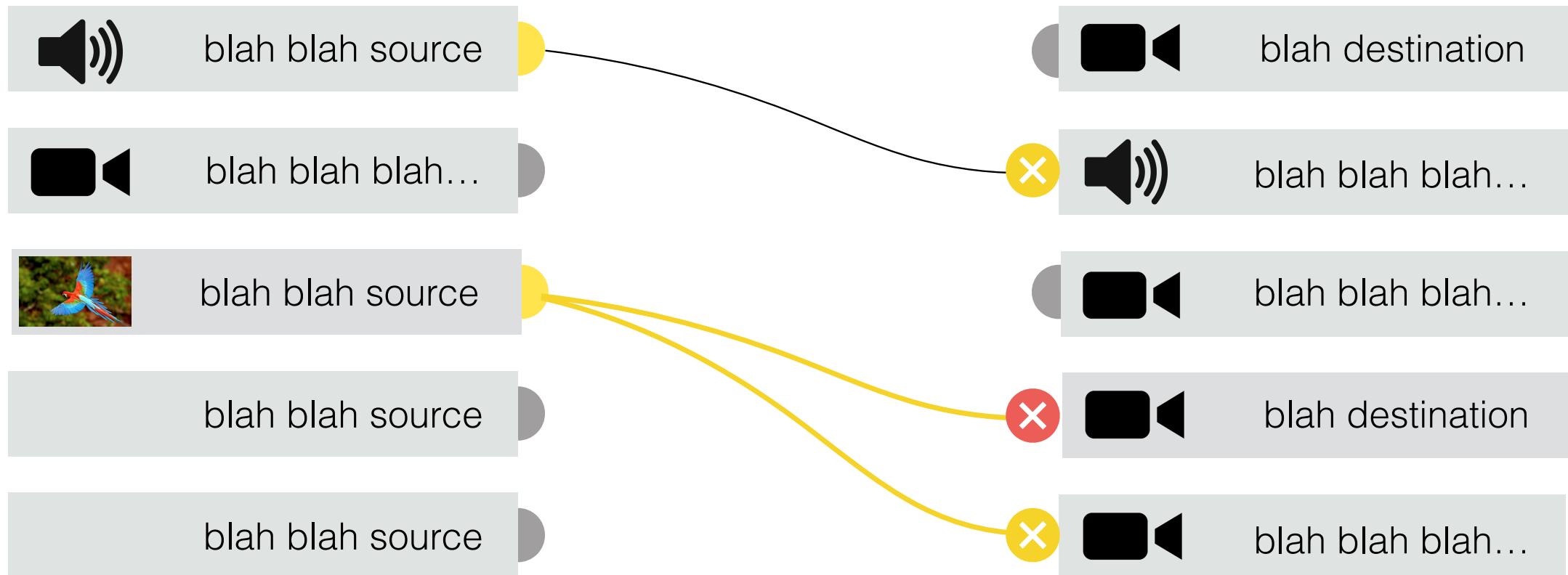




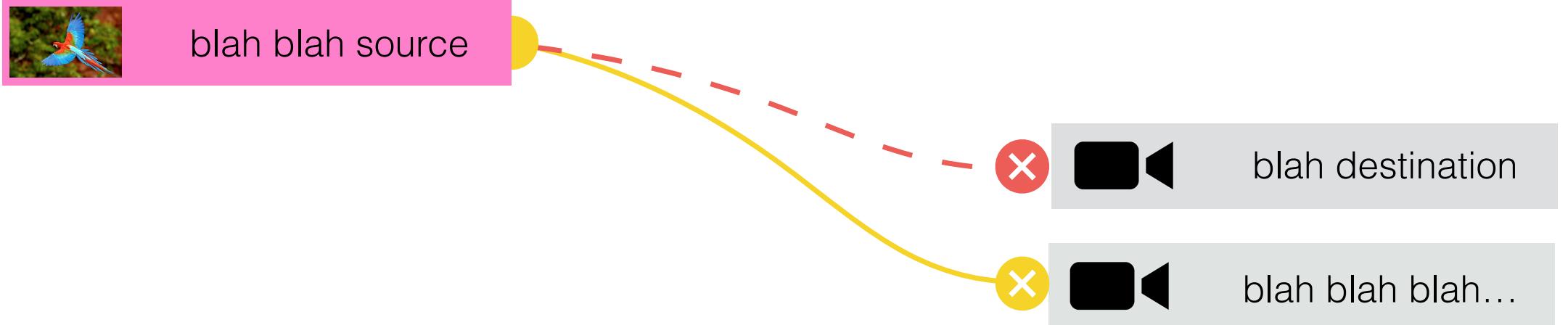


The user selects anywhere that isn't an active area for another control in order to deselect a selected sender. The sender returns to its place in the list and loses its highlight. All routes and receivers return to 100% opacity. Nodes of receivers incompatible with the previously selected sender become visible again.

# Deleting a Route



The user selects a crossed node for a routed destination by clicking or tapping it. The node turns red while the mouse / finger is down. The unroute action is triggered on mouse/finger up within the node's active area. If the user drags their finger/pointer out of the active area, the node returns to yellow to indicate that an unclick/touch-up event will have no effect.



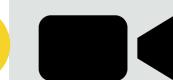
The app sends a 'destroy' call to the api. The UI tells the user that it is in the process of destroying the route by turning the line red. The line also becomes dashed. The source node reverts to the colour it would have been if the selected route had already been destroyed (in this case, yellow, as there is another route from that source).



blah blah source



blah destination



blah blah blah...

Once the destroy operation has been processed the line fades out slowly (~1s), the destroyed node moves back to its original position, and it remains red for ~ 2 seconds



blah blah source



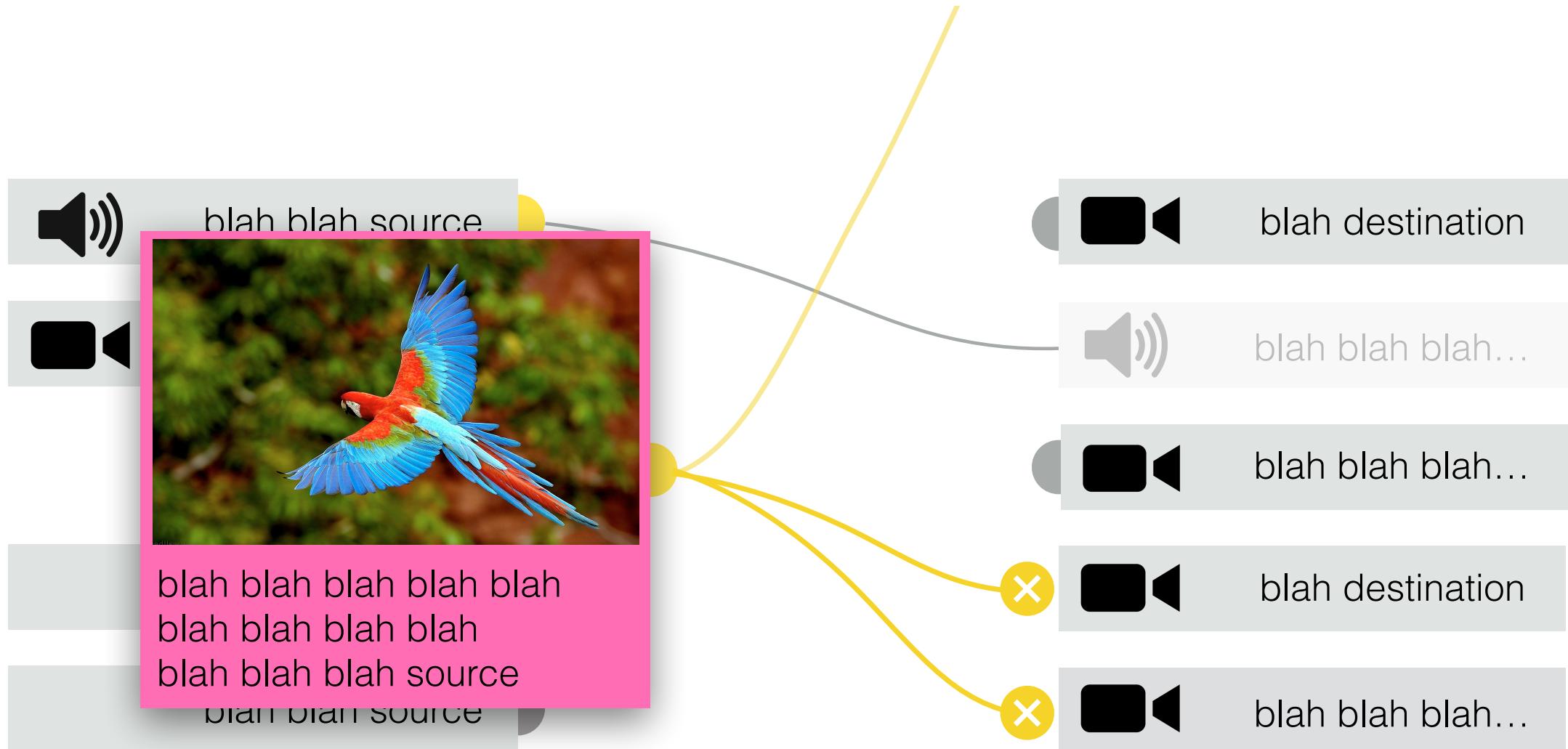
blah destination



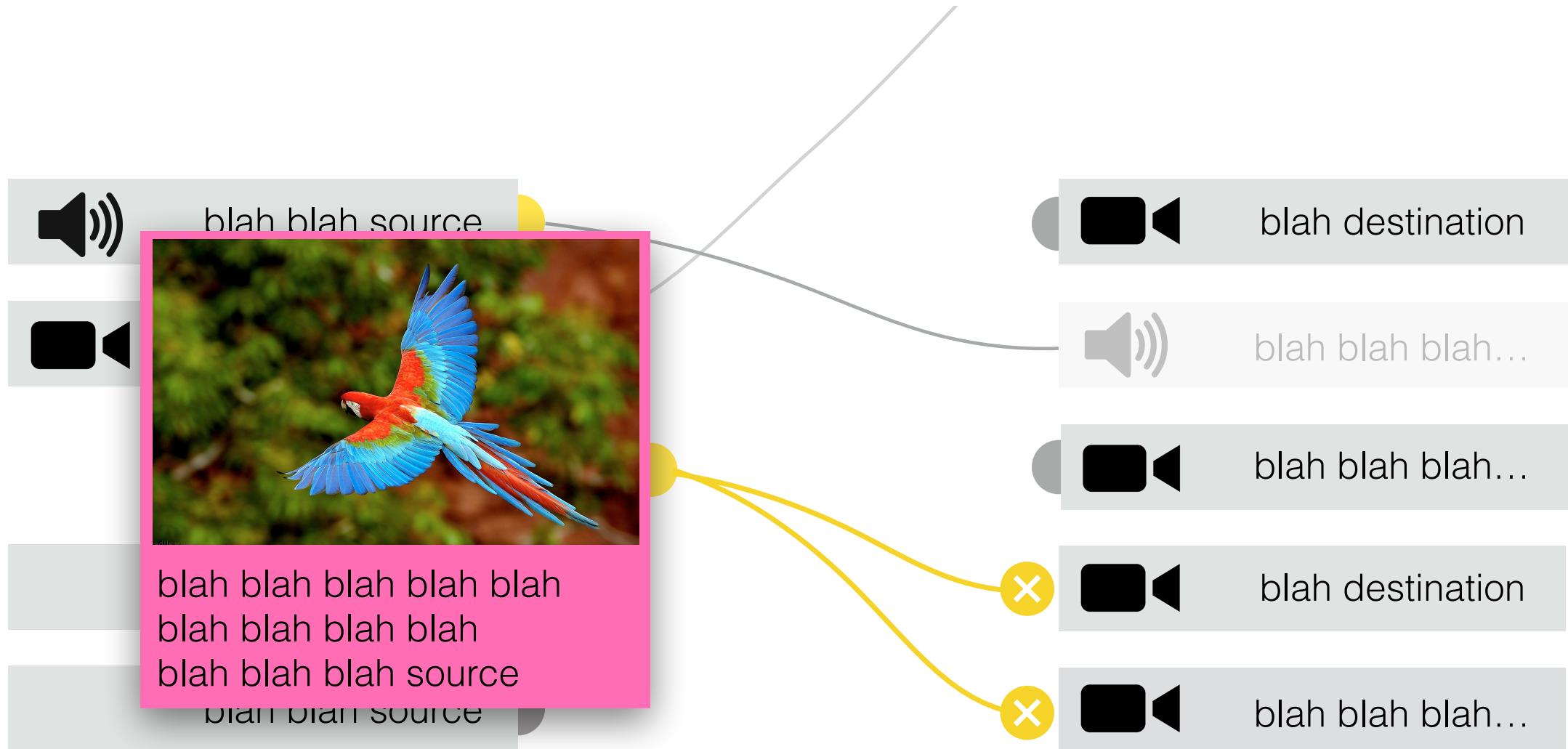
blah blah blah...

The nodes then lose their red colour, returning to whatever state they would otherwise have had. (The destination will always be grey after an unroute.)

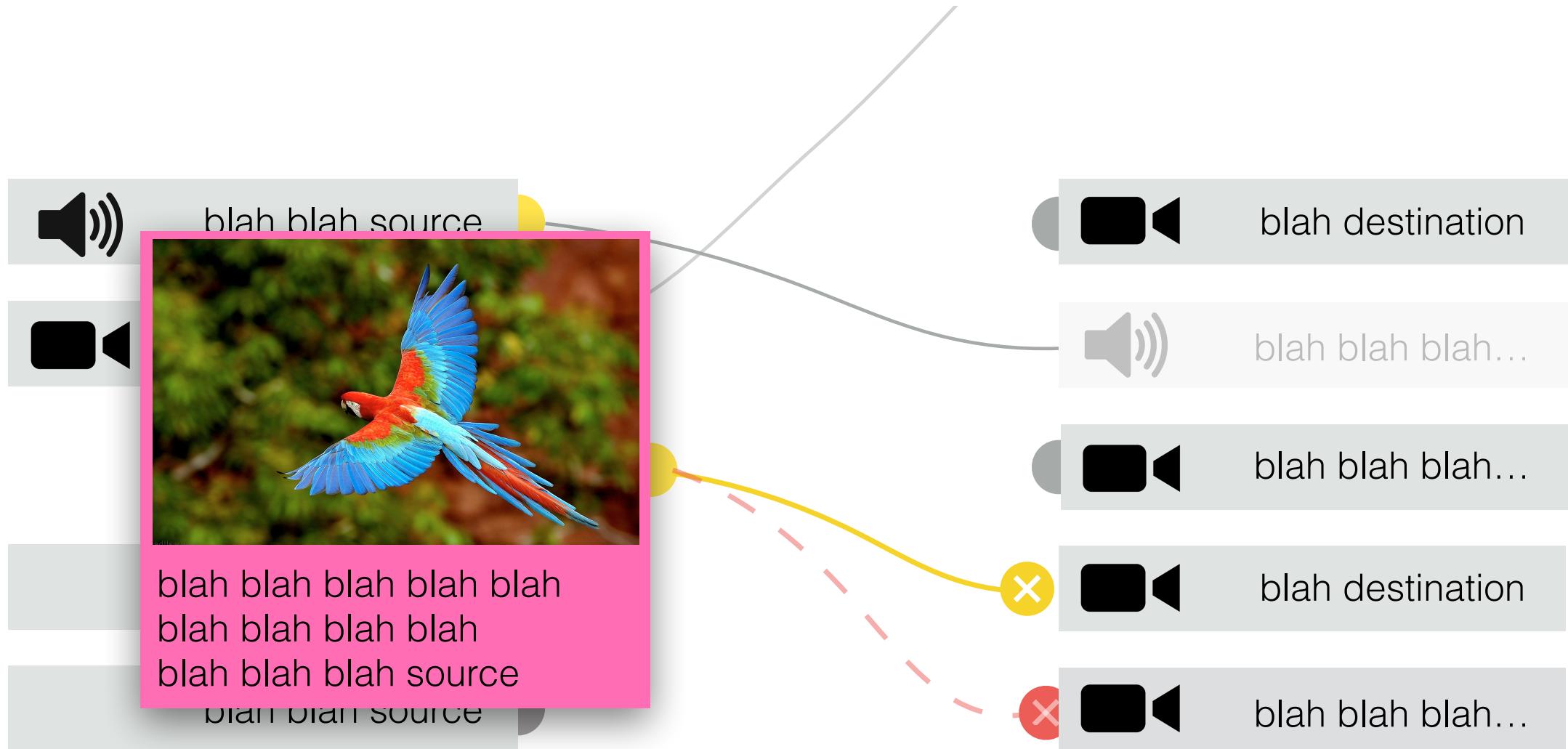
Notifications of  
changes



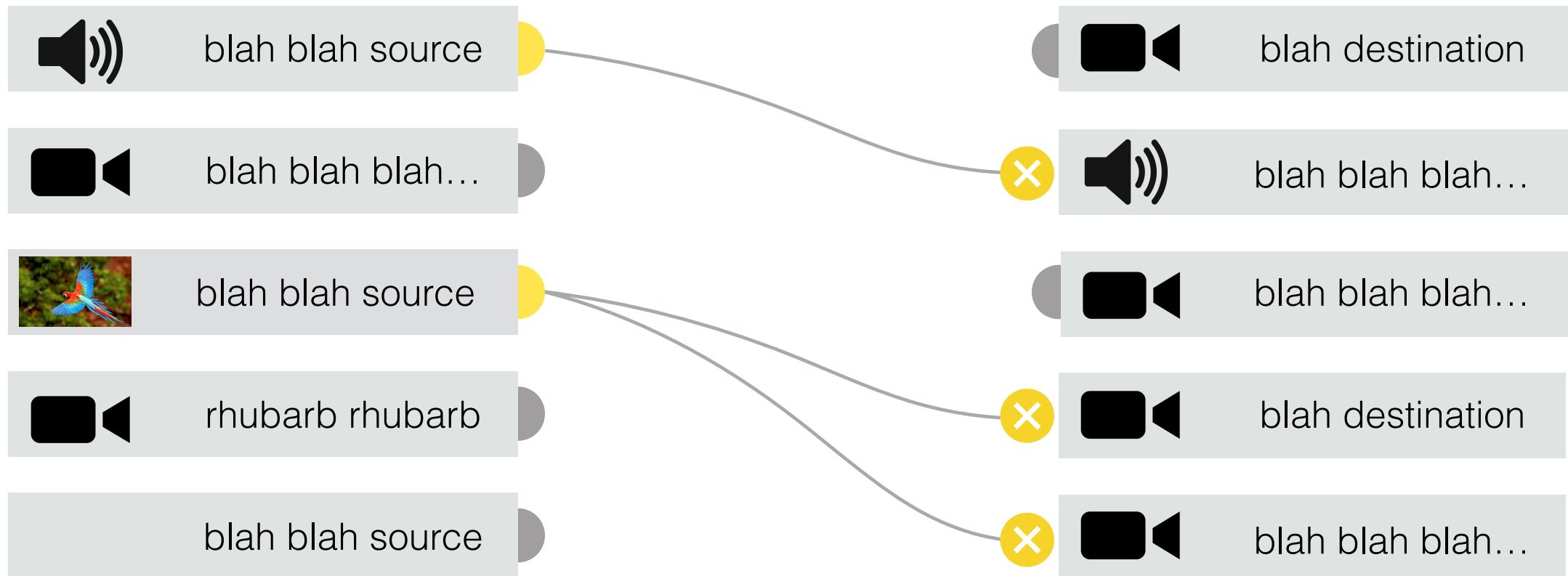
A notification is received from the subscription API that a new route is available. The UI indicates this to the user by adding it to the view, animating the addition with a slow fade (~1s) to the appropriate rendering state. In this case, the new route (shown half-way through the fade) is from the selected sender, so the fade is to a thick yellow line. If a notification is received that the route has been destroyed again before the animation has completed, the animation reverses and fades back out at the same speed.



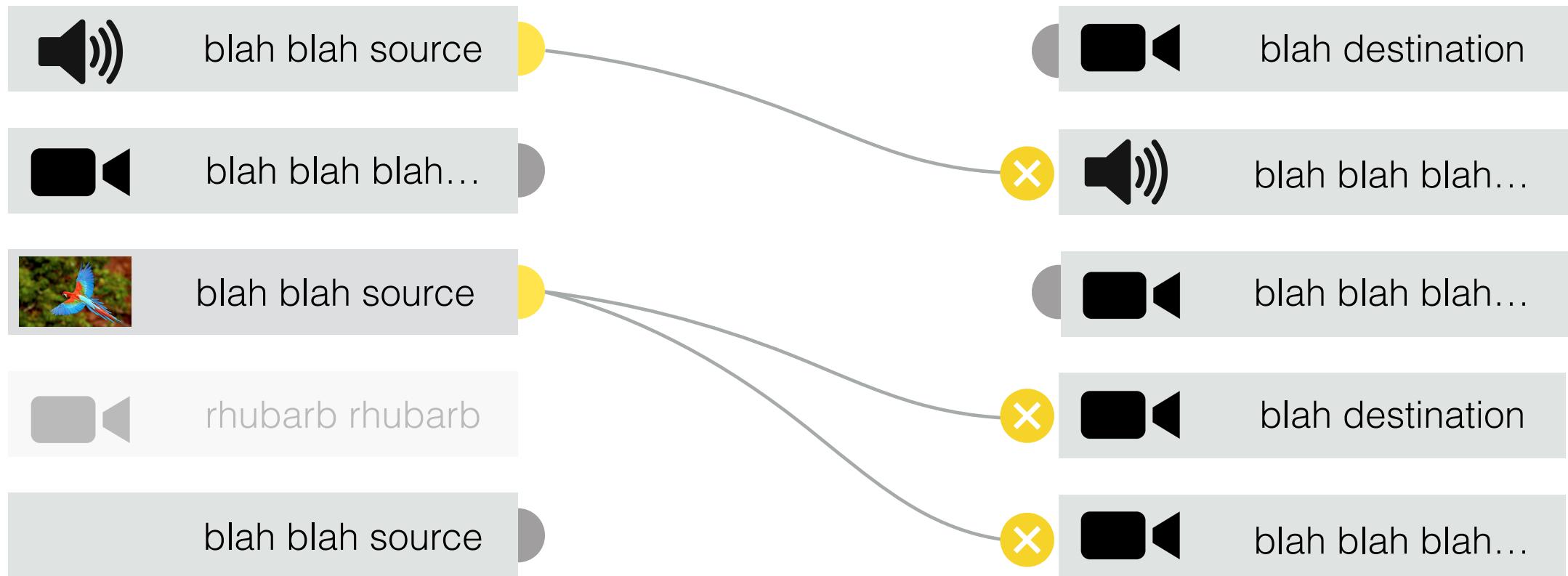
A notification is received from the subscription API that a route has been destroyed. The UI indicates this to the user by removing it from the view, animating the removal with a slow fade (~1s) to the appropriate rendering state. In this case, the destroyed route (shown half-way through the fade) is from an unselected sender, so the fade is from a thin grey line. If a notification is received that the route has been created again before the animation has completed, the animation reverses and fades back in again at the same speed.



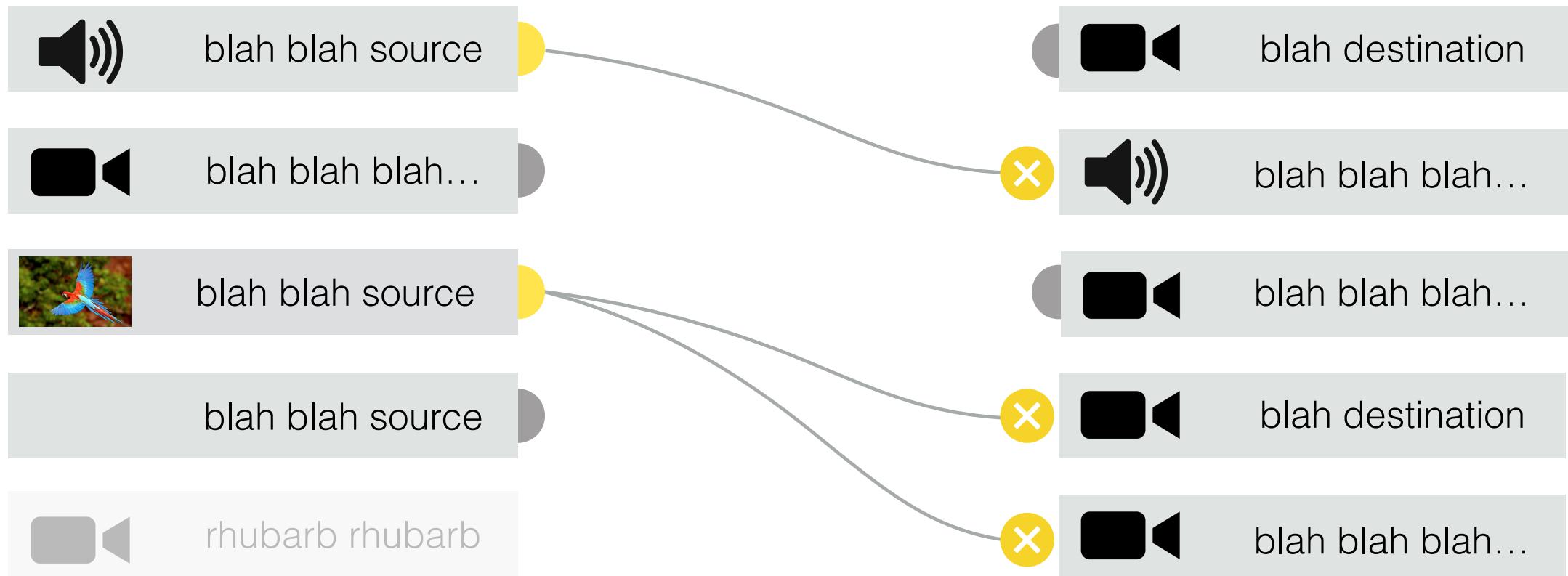
The user has requested that a route be destroyed, and a notification is received from the subscription API that this action has taken place (perhaps in response to the user request; perhaps coincidentally.) This is indicated to the user as if their request had been processed successfully, regardless of the cause of the change.



The application receives a notification that a new sender, “rhubarb rhubarb”, is available. It is inserted into the list at the appropriate position. The items below its position in the list translate to their new position (animate 0 25s) and the new sender then fades in (animate 0 5s). New receivers are treated the same way.

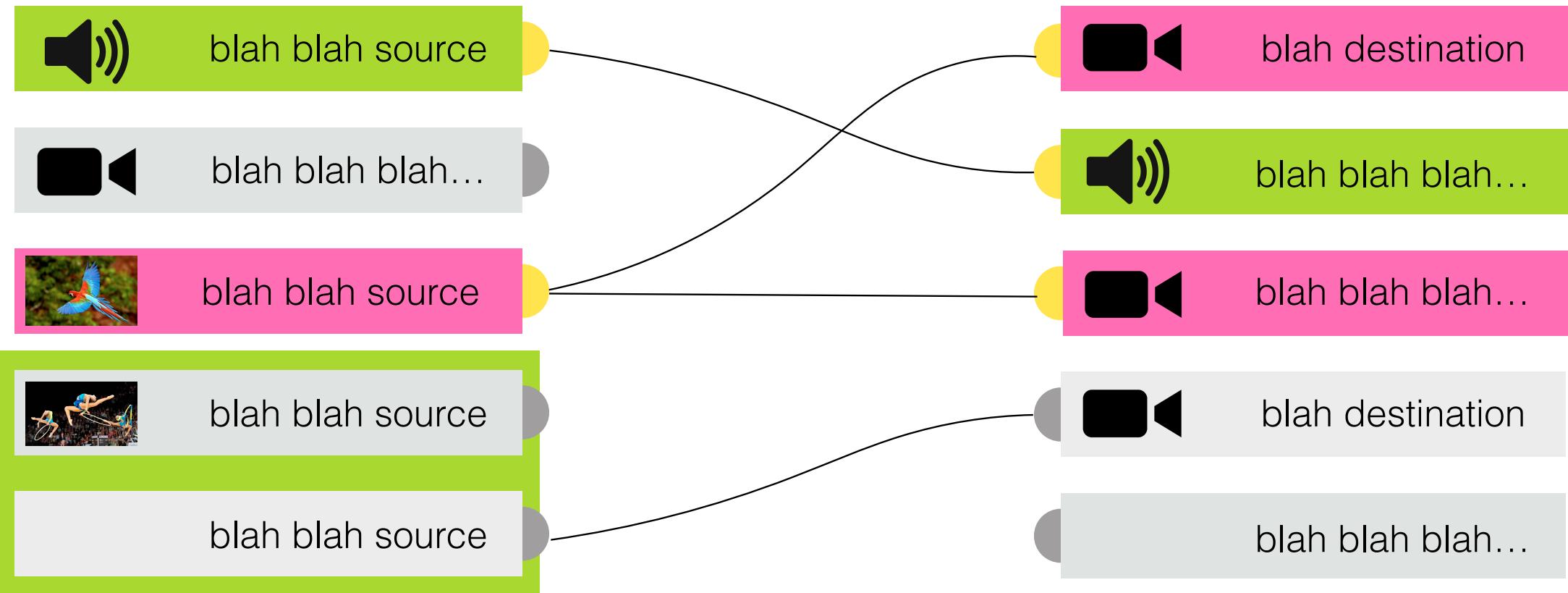


The application receives a notification that the sender “rhubarb rhubarb” is no longer available. It loses its node, and fades to 25% opacity; its routes fade out completely (0.25s). It can no longer be selected. In the event of a page refresh it is no longer shown. Unavailable receivers are treated the same way.



After one minute, the unavailable sender moves to the bottom of the list. The senders beneath its former position in the list move up to fill the gap. (Animate both, 0.5s.) It can not be selected. In the event of a page refresh, it is no longer shown. If more than one unavailable senders are in this state, they are sorted according to the order of the rest of the list (ie “unavailable for >1 minute”) becomes the first sort critereon. If the sender becomes available again, it moves back into its correct position in the list of available senders (animate, 0.5s) and regains its original styling. Receivers are treated the same way.

Ignore everything after  
this.





blah blah source



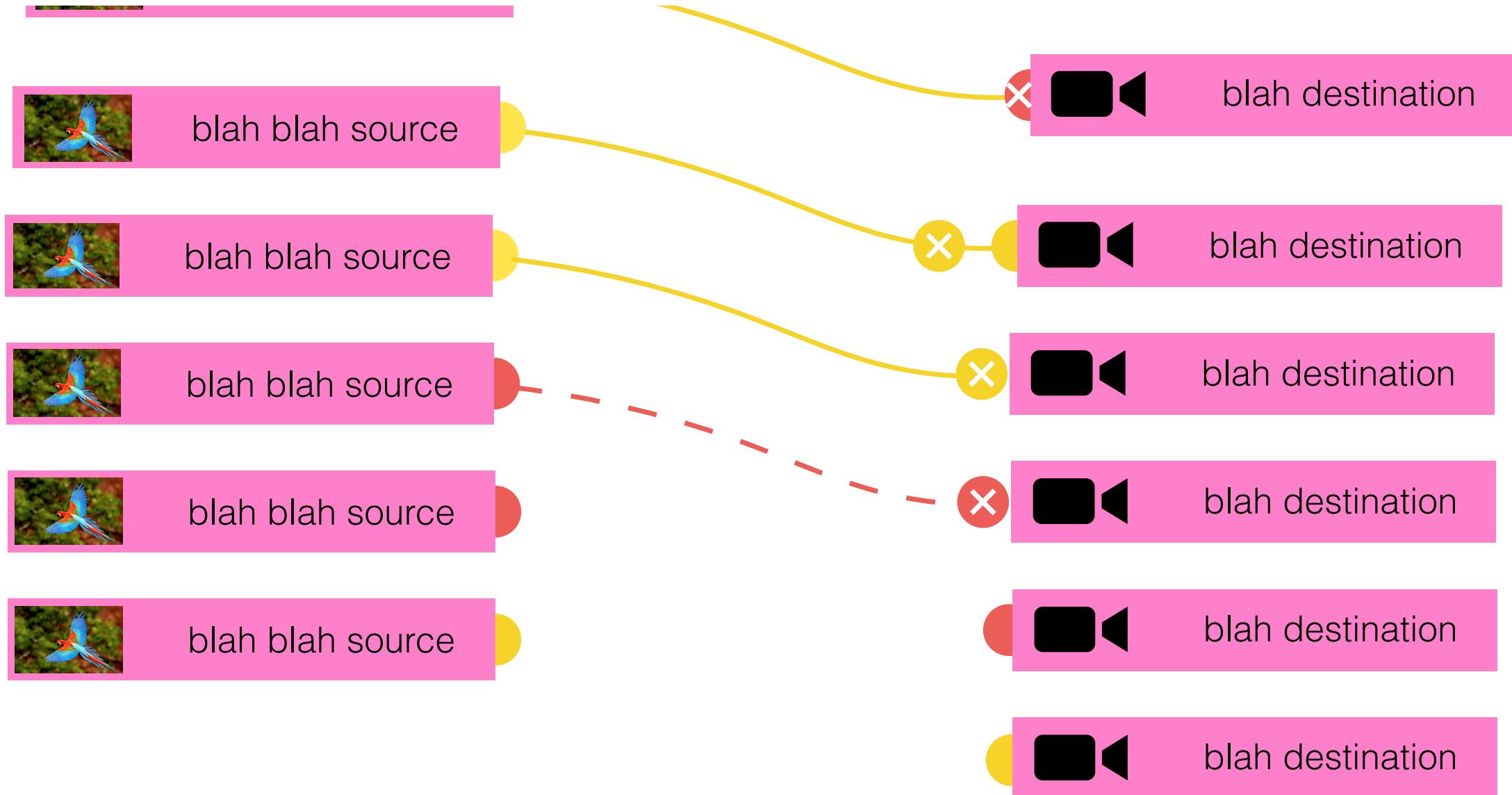
blah blah blah blah blah  
blah blah blah blah  
blah blah blah source



blah blah source



blah blah blah blah blah  
blah blah blah blah  
blah blah blah source



FIND

VIEW/  
SELECT

CONFIRM

## CONFIRM

