

## Kepler - Bug #5703

### GUI bug - Deleting an Actor with connected port causes a diamond relation to be created.

08/31/2012 01:49 PM - Derik Barseghian

<b>Status:</b>	Resolved	<b>Start date:</b>	08/31/2012
<b>Priority:</b>	Normal	<b>Due date:</b>	
<b>Assignee:</b>	Christopher Brooks	<b>% Done:</b>	0%
<b>Category:</b>	interface	<b>Estimated time:</b>	0.00 hour
<b>Target version:</b>	2.4.0	<b>Spent time:</b>	0.00 hour
<b>Bugzilla-Id:</b>	5703		
<b>Description</b>			
In the last week or so I've begun seeing a few GUI anomalies on trunk. One is:  Deleting an actor with a connected port causes a diamond relation to be created. If you undo the delete, the deleted actor returns, but the relation from actor1 to actor2 now connects via the diamond relation.  Replicate: Drag out a Constant and a Display. Connect output of Constant to input of Display. Delete Display Bug: diamond relation appears on canvas  This bug occurs in both kepler and vergil.			

#### History

##### #1 - 08/31/2012 04:20 PM - Christopher Brooks

I think is caused by one of these changes:

2012-08-23 21:43 eal

- [r64468] /trunk/diva/canvas/interactor/BasicGrabHandle.java: Removed overly aggressive prevention of NPE that also prevented a legitimate relation from occurring.

2012-08-23 21:42 eal

- [r64467] /trunk/diva/canvas/TransformContext.java: More tightly targetted prevention of a non-reproducible NPE upon dragging and connecting ports.

##### #2 - 08/31/2012 04:27 PM - Christopher Brooks

I think I was wrong about what causes this, more to follow.

##### #3 - 08/31/2012 07:50 PM - Christopher Brooks

The problem is reproducible after this change:

2012-08-18 17:08 eal

- [r64430] /trunk/ptolemy/actor/Manager.java,  
/trunk/ptolemy/actor/lib/hoc/LifeCycleManager.java,  
/trunk/ptolemy/kernel/undo/RedoChangeRequest.java,  
/trunk/ptolemy/kernel/undo/UndoChangeRequest.java,  
/trunk/ptolemy/kernel/util/Changeable.java,  
/trunk/ptolemy/kernel/util/NamedObj.java,  
/trunk/ptolemy/moml/MoMLParser.java: Fixed long standing deadlock that occurs while editing models while they run

Edward and I spent some time last week trying to reproduce the problem, but could not.

I can reproduce the problem by deleting the Display or the Const.

##### #4 - 09/07/2012 10:27 AM - Christopher Brooks

Fixed by Edward:

I've checked in a change that I hope fixes the problem we were chasing today. The cause is interesting...

When we edit a model, the editor constructs a MoML string for the edit actions. E.g., to remove a Display actor, it says:

```
&lt;deleteEntity name="Display"/&gt;
```

It passes this string to a MoMLChangeRequest, and calls requestChange() to request that the change be executed.

If the model is not running, then the change request will be executed immediately within the requestChange() method.

However, executing a change request may result in additional changes being requested... E.g., an additional change may be requested to remove a relation.

Before my checkin, that second change request was NOT being executed immediately. It would just sit in a queue until it was time to execute another change request (e.g., move an actor).

The new code in NamedObj is now also simpler:

```
public void requestChange(ChangeRequest change) {
    NamedObj container = getContainer();

    if (container != null) {
        container.requestChange(change);
    } else {
        // Synchronize to make sure we don't modify
        // the list of change requests while some other
        // thread is also trying to read or modify it.
        synchronized (_changeLock) {
            // Queue the request.
            // Create the list of requests if it doesn't already exist
            if (_changeRequests == null) {
                _changeRequests = new LinkedList<ChangeRequest>();
            }

            _changeRequests.add(change);
        }
        if (!_deferChangeRequests) {
            executeChangeRequests();
        }
    }
}
```

**#5 - 03/27/2013 02:31 PM - Redmine Admin**

Original Bugzilla ID was 5703