GEF – Figure – Two Traces
Use Full Trace (No Slicing)

Supporting Applications : [Flow (Activity), Shapes (Rectangle)]

```
import org.eclipse.gef.editpolicies.GraphicalNodeEditPolicy;
import org.eclipse.draw2d.IFigure;
import org.eclipse.jface.action.IAction;
import org.eclipse.gef.EditPartFactory;
import org.eclipse.gef.palette.PaletteDrawer;
import org.eclipse.gef.requests.CreateRequest;
import org.eclipse.gef.palette.PaletteRoot;
import org.eclipse.gef.NodeEditPart;
import org.eclipse.gef.requests.GroupRequest;
import org.eclipse.gef.ui.actions.RedoRetargetAction;
import org.eclipse.gef.RootEditPart;
import org.eclipse.jface.resource.ImageDescriptor;
import org.eclipse.gef.EditPartViewer;
import org.eclipse.gef.palette.ToolEntry;
import org.eclipse.gef.ui.parts.GraphicalViewerKeyHandler;
import org.eclipse.gef.palette.PaletteSeparator;
import org.eclipse.gef.dnd.TemplateTransferDropTargetListener;
import org.eclipse.gef.editparts.AbstractGraphicalEditPart;
import org.eclipse.gef.palette.MarqueeToolEntry;
import org.eclipse.gef.ui.actions.ActionRegistry;
import java.util.EventObject;
import org.eclipse.gef.palette.ConnectionCreationToolEntry;
import org.eclipse.gef.ui.actions.DeleteRetargetAction;
import java.util.List;
import org.eclipse.gef.ContextMenuProvider;
import org.eclipse.gef.requests.CreationFactory;
import org.eclipse.gef.ui.actions.ActionBarContributor;
import org.eclipse.ui.IEditorInput;
import org.eclipse.ui.IWorkbenchPartSite;
import org.eclipse.gef.editpolicies.ComponentEditPolicy;
import org.eclipse.gef.palette.CombinedTemplateCreationEntry;
import org.eclipse.ui.IEditorPart;
import org.eclipse.gef.dnd.TemplateTransferDragSourceListener;
import org.eclipse.swt.dnd.Transfer;
import org.eclipse.gef.editpolicies.RootComponentEditPolicy;
import org.eclipse.ui.IWorkbenchPart;
import org.eclipse.gef.DefaultEditDomain;
import org.eclipse.ui.part.WorkbenchPart;
import org.eclipse.gef.ui.parts.GraphicalEditor;
import org.eclipse.gef.editpolicies.LayoutEditPolicy;
import org.eclipse.gef.commands.Command;
import org.eclipse.gef.EditPart;
import org.eclipse.gef.palette.PaletteGroup;
import org.eclipse.gef.ui.actions.UndoRetargetAction;
import org.eclipse.gef.GraphicalViewer;
import org.eclipse.gef.requests.SimpleFactory;

public class AppLayoutEditPolicy
extends LayoutEditPolicy {

      public Command getCreateCommand(CreateRequest) {
            Object object1 = CreateRequest.getNewObject();  // REPEATED!
            AppCommand appCommand = new AppCommand();
            EditPart editPart = appComponentEditPolicy.getHost(); // REPEATED!
            Object object = editPart.getModel();      // REPEATED!
      }
}

public class AppGraphicalNodeEditPolicy
extends GraphicalNodeEditPolicy {
}

public class AppComponentEditPolicy
extends ComponentEditPolicy {

      public Command createDeleteCommand(GroupRequest) {
            AppCommand appCommand = new AppCommand();
            EditPart editPart = appComponentEditPolicy.getHost(); // REPEATED!
            Object object = editPart1.getModel();     // REPEATED!
            EditPart editPart1 = editPart.getParent();
      }
}

public class AppAbstractGraphicalEditPart
implements NodeEditPart
extends AbstractGraphicalEditPart {

      public IFigure createFigure() {
      }

      public void createEditPolicies() {
            AppGraphicalNodeEditPolicy appGraphicalNodeEditPolicy = new AppGraphicalNodeEditPolicy();
            RootComponentEditPolicy rootComponentEditPolicy = new RootComponentEditPolicy();
            AppComponentEditPolicy appComponentEditPolicy = new AppComponentEditPolicy();
            appAbstractGraphicalEditPart.installEditPolicy(rootComponentEditPolicy && appGraphicalNodeEditPolicy && appComponentEditPolicy);    // REPEATED!
      }

      public void deactivate() {
            Object object = appAbstractGraphicalEditPart.getModel();    // REPEATED!
      }

      public List getModelChildren() {
            Object object = appAbstractGraphicalEditPart.getModel();    // REPEATED!
      }

      public void activate() {
            Object object = appAbstractGraphicalEditPart.getModel();    // REPEATED!
      }

      public List getModelTargetConnections() {
            Object object = appAbstractGraphicalEditPart.getModel();    // REPEATED!
      }

      public void refreshVisuals() {
            Object object = appAbstractGraphicalEditPart.getModel();    // REPEATED!
            IFigure ifigure = appAbstractGraphicalEditPart.getFigure(); // REPEATED!
      }

      public List getModelSourceConnections() {
            Object object = appAbstractGraphicalEditPart.getModel();    // REPEATED!
      }
}

public class AppditPartFactory
implements EditPartFactory  {

      public EditPart createEditPart(object1 && appAbstractGraphicalEditPart && editPart) {
```

```
            AppAbstractGraphicalEditPart appAbstractGraphicalEditPart = new AppAbstractGraphicalEditPart();
            appAbstractGraphicalEditPart.setModel(object1); // REPEATED!
      }
}

public class AppGraphicalEditor
extends GraphicalEditor {

      public void commandStackChanged(EventObject) {
            WorkbenchPart.firePropertyChange(int);
      }

      public void setInput(IEditorInput) {
      }

      public void configureGraphicalViewer() {
            AppditPartFactory appditPartFactory = new AppditPartFactory();
            /*                      Cyclic Statements                       */
            ActionRegistry actionRegistry = GraphicalEditor.getActionRegistry();    // MAY REPEAT!
            IWorkbenchPartSite iworkbenchPartSite = IWorkbenchPart.getSite(); // REPEATED!
            graphicalViewer.setContextMenu(appContextMenuProvider);
            graphicalViewer.setKeyHandler(graphicalViewerKeyHandler);
            GraphicalViewer graphicalViewer = GraphicalEditor.getGraphicalViewer(); // MAY REPEAT!
            GraphicalViewerKeyHandler graphicalViewerKeyHandler = new GraphicalViewerKeyHandler(graphicalViewer);
            graphicalViewer.setRootEditPart(RootEditPart);
            graphicalViewer.setEditPartFactory(appditPartFactory);
            AppContextMenuProvider appContextMenuProvider = new AppContextMenuProvider(graphicalViewer && actionRegistry);
      }

      public void initializeGraphicalViewer() {
            GraphicalViewer graphicalViewer = GraphicalEditor.getGraphicalViewer(); // REPEATED!
            graphicalViewer.setContents(EditPart):()||(Object):();
            TemplateTransferDropTargetListener templateTransferDropTargetListener = new TemplateTransferDropTargetListener(graphicalViewer);
            graphicalViewer.addDropTargetListener(templateTransferDropTargetListener);
      }
}

public class AppActionBarContributor
extends ActionBarContributor {

      public void declareGlobalActionKeys() {
      }

      public void buildActions() {
            DeleteRetargetAction deleteRetargetAction = new DeleteRetargetAction();
            RedoRetargetAction redoRetargetAction = new RedoRetargetAction();
            UndoRetargetAction undoRetargetAction = new UndoRetargetAction();
            ActionBarContributor.addRetargetAction(deleteRetargetAction && redoRetargetAction && undoRetargetAction);   // REPEATED!
      }
}

public class AppCommand
extends Command {

      public void execute() {
            editPart2.refreshChildren();
      }

}

public class AppContextMenuProvider
extends ContextMenuProvider {
}

public class SomeClass {

      public void someMethod() {
            IAction iaction = ActionBarContributor.getAction(String);   // REPEATED!
            MarqueeToolEntry marqueeToolEntry = new MarqueeToolEntry(String)||(String,String)||();
            DefaultEditDomain defaultEditDomain = new DefaultEditDomain(IEditorPart);
            GraphicalEditor.setEditDomain(defaultEditDomain);
            TemplateTransferDragSourceListener templateTransferDragSourceListener = new TemplateTransferDragSourceListener(EditPartViewer)||(EditPartViewer,Transfer);
            EditPartViewer.addDragSourceListener(templateTransferDragSourceListener);
            SimpleFactory simpleFactory = new SimpleFactory(Class);
            CombinedTemplateCreationEntry combinedTemplateCreationEntry = new CombinedTemplateCreationEntry(simpleFactory);
            PaletteGroup paletteGroup = new PaletteGroup(String,List)||(String);
            ConnectionCreationToolEntry connectionCreationToolEntry = new ConnectionCreationToolEntry(String,String,CreationFactory,ImageDescriptor,ImageDescriptor));
            PaletteDrawer paletteDrawer = new PaletteDrawer(String)||(String,ImageDescriptor);
            PaletteRoot paletteRoot = new PaletteRoot();
            paletteRoot.setDefaultEntry(ToolEntry);
            PaletteSeparator paletteSeparator = new PaletteSeparator()||(String);
      }
}
```