1. The first step in the morphing procedure will be to get the source and target polygons. This will be done by defining a point (probably the middle of canvas) as the common middle of the star shaped polygons. The middle point will be marked and the user will click around the point carefully to define the first star shaped polygon, connecting it at the end. This will generate the source polygon to be displayed and then the user will click around the once more to define the target polygon morph the source into. After the polygons are defined it is time to carry out the morphing.

To Morph the source polygon to the target polygon we will first interpolate lines coming from the common center point moving through each vertex of the source and target polygons. These lines intersect both the source and polygons and we will note where these intersections are. Next we will define a new polygon with the intersections of the source polygon as well as the source polygon vertices as the new polygon’s vertices. Now to achieve morphing we slowly move each vertex on the new polygon to the corresponding point on the target polygon incrementally. A point corresponds to a point on the target polygon if they lie on the same line that was interpolated from the common center. Once the polygon’s points have reached their destination point the morphing is complete.

1. The ADT needed for the morphing algorithm I will call starPoly. starPoly has the following data: a vector, center, that contains the x, y values of the center of a star polygon that starPoly represents; a vector, vertices, that contains all of the x,y values of the star polygon that starPoly represents; a vector, first, that contains the x, y values of the first vertex defined in the starPoly; and a Boolean value, isComplete, that is false until the starPoly is complete (start points and end points are the same ). Operations on a starPoly are as follows: newStarPoly(x,y) clears the current starPoly and populates it with the first vertex being x,y; populate(x,y) will add a vertex to the starPoly and if x,y are the same as the first vertex then isComplete is set to true; getVertex(n) will return the nth vertex in the starPoly (which are in order in the vector called vertices); setVertex(n, x,y) will set the nth vertex to the point x,y.
2. The algorithm to complete the morph will be as follows:  
   starPoly source, destination, working, modifiedDest;

vector point;

//construct source

Point=user click

Source.newStarPoly(point)

Point=user click

While(point!=source.first)

Source.populate(point)

Point=user click

//construct destination

Point=user click

destination.newStarPoly(point)

Point=user click

While(point!=destination.first)

destination.populate(point)

Point=user click

//interpolate

Working.newStarPoly(source.getVertex(0))

for each interpolated intersection on source and each source vertex in counter clockwise order:

working.populate(point)

modifiedDest.newStarPoly(desg.getVertex(0))

for each interpolated intersection on destination and each destination vertex in counter clockwise order:

modifiedDest.populate(point)

for 20 steps

for each vertex in working

working.setVertex(n, working.getVertex(n)+5% of the total distance from modifiedDest.getVertex(n))

stall to allow morphing to be seen.

repaint();

After 20 steps the morphing should be complete;

4). No other classes seem to be needed, perhaps a class that helps in determining the intersection parts.