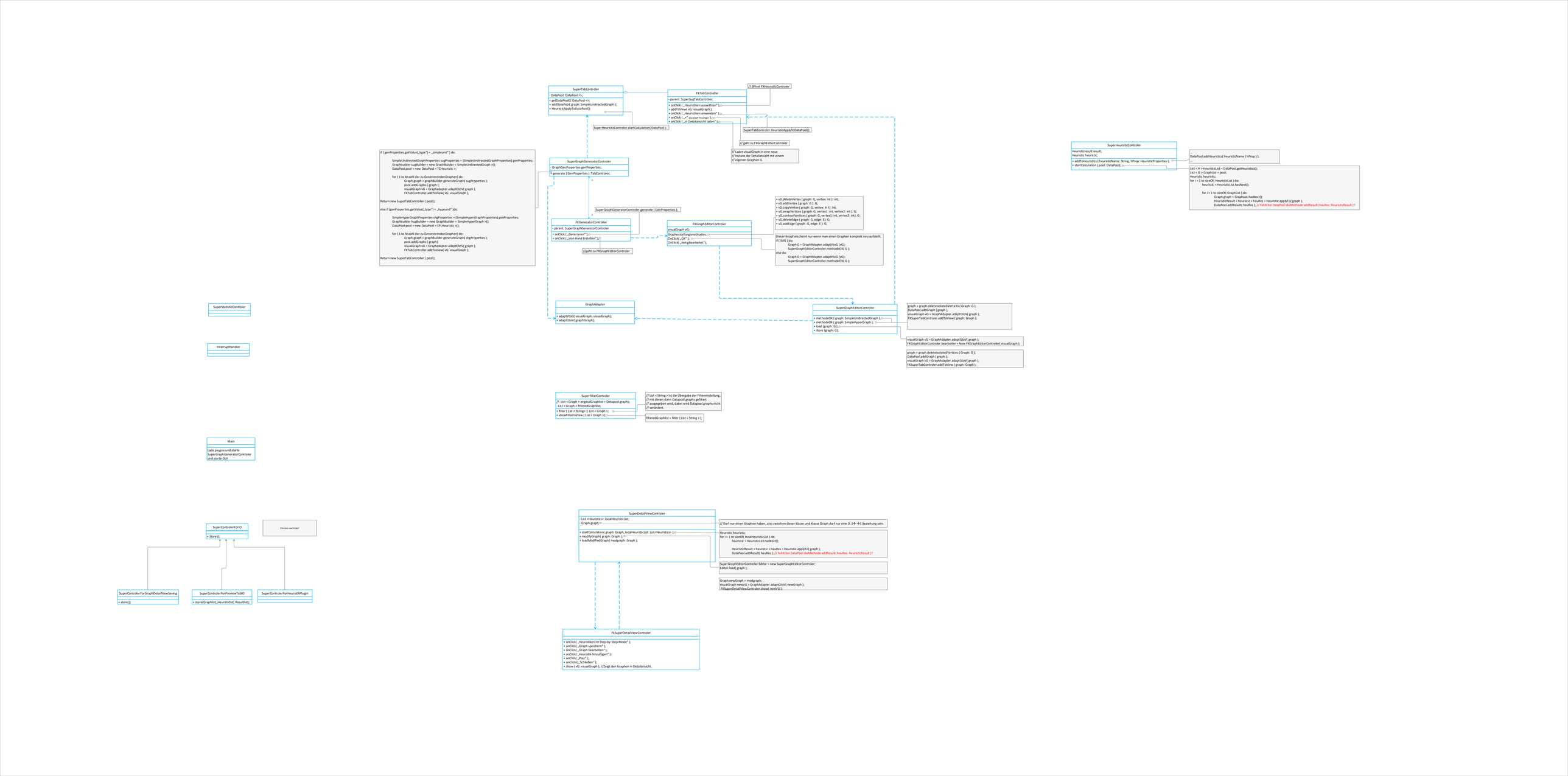
SuperTabControler
- DataPool: DataPool <V>;
+ getDataPool(): DataPool <V>;
+ addDataPool(graph: SimpleUndirectedGraph);
+ HeuristicApplyToDataPool();
FXTabController
- parent: SuperSugTabController;
+ onClick („Heuristiken auswählen");
+ addTo(newG: visualGraph);
+ onClick („Heuristiken anwenden");
+ onClick („+" nur bei Heuristiken);
+ onClick („in Detailansicht laden");
// Offnet FXHeuristicControler
SuperTabControler.HeuristicApplyToDataPool();
// geht zu FXGraphEditorControler
// Ladet visualGraph in eine neue
// Instanz der Detailansicht mit einem
// eigenen Graphen G.
SuperHeuristicControler.startCalculation(DataPool);
if (genProperties.getValue(_type") = „simpleund") do:
SimpleUndirectedGraphProperties sugProperties = (SimpleUndirectedGraphProperties) genProperties;
GraphBuilder sugBuilder = new GraphBuilder < SimpleUndirectedGraph >();
DataPool pool = new DataPool < TCHeuristic >;
for (1 to Anzahl der zu Generierenden Graphen) do:
Graph graph = graphBuilder.generateGraph(sugProperties);
pool.addGraphs (graph);
visualGraph vG = GraphAdapter.adaptGtoV(graph);
FXTabController.addToView(vG: visualGraph);
Return new SuperTabControler (pool);
else if (genProperties.getValue(_type") = „hypeund")do:
SimpleHyperGraphProperties shgProperties = (SimpleHyperGraphProperties.) genProperties;
GraphBuilder hugBuilder = new GraphBuilder < SimpleHyperGraph >();
DataPool pool = new DataPool < EFLHeuristic >();
for (1 to Anzahl der zu Generierenden Graphen) do:
Graph graph = graphBuilder.generateGraph(shgProperties);
pool.addGraphs (graph);
visualGraph vG = GraphAdapter.adaptGtoV(graph);
FXTabController.addToView(vG: visualGraph);
Return new SuperTabControler (pool);
SuperGraphGeneratorControler
- GraphGenProperties genProperties;
+ generate (GenProperties): TabController;
SuperGraphGeneratorControler.generate (GenProperties);
FXGeneratorControler
- parent: SuperGraphGeneratorControler
+ onClick („Generieren");
+ onClick („Von Hand Erstellen");
//geht zu FXGraphEditorControler
FXGraphEditorControler
visualGraph vG;
Graphenstellungsmethoden;
OnClick(_„OK");
OnClick(_„Fertig Bearbeitet");
+ vG.deleteVertex (graph: G, vertex: int): int;
+ vG.addVertex (graph: G): G;
+ vG.copyVertex (graph: G, vertex: in t): int;
+ vG.swapVertices (graph: G, vertex1: int, vertex2: int): G;
+ vG.contractVertices (graph: G, vertex1: int, vertex2: int): G;
+ vG.deleteEdge (graph: G, edge: E): G;
+ vG.addEdge (graph: G, edge: E): G;
Dieser Knopf erscheint nur wenn man einen Graphen komplett neu aufstellt.
if (SLVG) do:
Graph G = GraphAdapter.adaptVtoG (vG);
SuperGraphEditorControler.methodeOK(G);
else do:
Graph G = GraphAdapter.adaptVtoG (vG);
SuperGraphEditorControler.methodeOK(G);
GraphAdapter
+ adaptVtoG(visualGraph: visualGraph);
+ adaptGtoV(graphGraph);
SuperGraphEditorControler
+ methodeOK (graph: SimpleUndirectedGraph);
+ methodeOK (graph: SimpleHyperGraph);
+ load (graph: G);
+ store (graph: G);
graph = graph.deleteIsolatedVertices (Graph: G);
DataPool.addGraph (graph);
visualGraph vG = GraphAdapter.adaptGtoV(graph);
FXSuperTabControler.addToView (graph: Graph);
visualGraph vG = GraphAdapter.adaptGtoV(graph);
FXGraphEditorControler bearbeiter = New FXGraphEditorControler(visualGraph);
graph = graph.deleteIsolatedVertices (Graph: G);
DataPool.addGraph (graph);
visualGraph vG = GraphAdapter.adaptGtoV(graph);
FXSuperTabControler.addToView (graph: Graph);
SuperHeuristicControler
HeuristicResult result;
Heuristic heuristic;
+ addToHeuristics (heuristicName: String, hProp: HeuristicProperties);
+ startCalculation (pool: DataPool);
DataPool.addHeuristic(heuristicName (hProp));
List < H > HeuristicList = DataPool.getHeuristics();
List < G > GraphList = pool;
Heuristic heuristic;
for i = 1 to sizeOf(HeuristicList) do:
heuristic = HeuristicList.hasNext();
for i = 1 to sizeOf(GraphList) do:
Graph graph = GraphList.hasNext();
HeuristicResult = heuristic > heuRes = Heuristic.applyTo(graph);
DataPool.addResult(heuRes); // Fehlt bei DataPool derMethode addResult(heuRes: HeuristicResult)?
SuperStatistixControler
InterruptHandler
Main
Lade plugins und starte
SuperGraphGeneratorControler
und starte GUI
SuperFilterControler
// - List < Graph > originalGraphList = DataPool.graphs;
- List < Graph > filteredGraphList;
+ filter (List < String>): List <Graph >;
+ showFilterInView (List < Graph >);
// List < String > ist die Übergabe der Filtereinstellung,
// mit denen dann Datapool.graphs gefiltert
// ausgegeben wird, dabei wird Datapool.graphs nicht
// verändert.
filteredGraphList = filter (List < String >);
SuperControlerForIO
+ Store ();
Christian macht das!
SuperControlerForGraphDetailViewSaving
+ store();
SuperControlerForPreviewTabIO
+ store(GraphList, HeuristicList, ResultList);
SuperControlerForHeuristikPlugin
SuperDetailViewControler
- List <HeuristicS> localHeuristicList;
- Graph graph;
+ startCalculation(graph: Graph, localHeuristicList: List<HeuristicS>);
+ modifyGraph(graph: Graph);
+ loadModifiedGraph(modgraph: Graph);
// Darf nur einen Graphen haben, also zwischen dieser klasse und Klasse Graph darf nur eine 0..1 ←→1 Beziehung sein.
Heuristic heuristic;
for i = 1 to sizeOf(localHeuristicList) do:
heuristic = HeuristicList.hasNext();
HeuristicResult < heuristic > heuRes = HeuristicS.applyTo(graph);
DataPool.addResult(heuRes); // Fehlt bei DataPool derMethode addResult(heuRes: HeuristicResult)?
SuperGraphEditorControler Editor = new SuperGraphEditorControler;
Editor.load(graph);
Graph newGraph = modgraph;
visualGraph newVG = GraphAdapter.adaptGtoV(newGraph);
FXSuperDetailViewController.show(newVG);
FXSuperDetailViewControler
+ onClick(„Heuristiken im Step-by-Step-Mode");
+ onClick(„Graph speichern");
+ onClick(„Graph bearbeiten");
+ onClick(„Heuristik hinzufügen");
+ onClick(„Play");
+ onClick(„Schließen");
+ show (vG: visualGraph); //Zeigt den Graphen in Detailansicht.