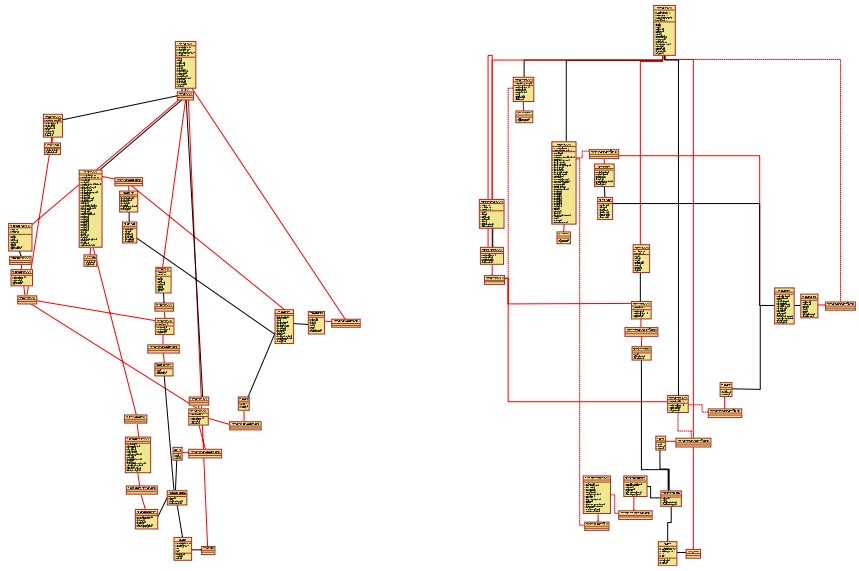

(a) symmetric style

(b) orthogonal style

Figure 4: Industrial layout in different styles. Same diagram as in Figure 2.

clearly identify for every edge the connected endpoints or to identify the inheritance hierarchies. We leave it to the reader to find out how many inheritance hierarchies are contained in both diagrams. In Section 5, Figure 17 and Figure 18, the same diagrams are given with GoVisual layout in which the number of inheritance hierarchies is clearly visible.

Figures 4(a) and 4(b) demonstrate that pure non-hierarchical visualization must fail for class diagrams. Both Figures show the same graph as in Figure 2 and have been created by one of the leading software development tools. The layout given in Figure 4(a) is a typical symmetric layout that is the result of a low budget solution and leaves the human reader uninformed about the structure of the software project. The orthogonal layout of the same diagram demonstrates, apart from the obvious technical weaknesses, that the missing hierarchical information still does not reveal clear information on the project.

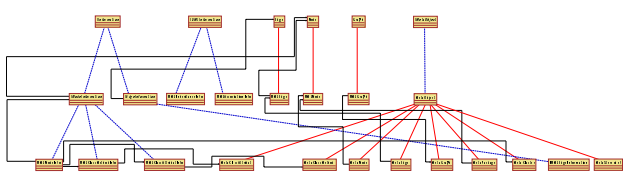

Figure 5: Seemann style layout

The weakness of strictly hierarchical or strictly non-hierarchical layout algorithms for mixed-hierarchical graphs has been recognized early by [Seemann 1997], presenting a two phase method combining Sugiyama layout [Sugiyama et al. 1981] for the inheritance hierarchies and a routing method for the undirected edges. The approach has been enhanced by [Eichelberger 2002]. The drawback of this method is immediate: routing edges within a given drawing results in hard combinatorial problems that are attacked via local optimization methods. With an increasing number of undirected edges, this approach leads to diagrams which are difficult to read. Figure 5 shows a layout of a diagram with one inheritance hierarchy produced by this approach given in [Eichelberger 1999].

One combinatorial subproblem that occurs when drawing mixed-hierarchical graphs, namely mixed upward planarization that we shall discuss in Section 3.3, has also been considered in [Eiglsperger and Kaufmann 2001].

1.3 The GoVisual Approach

Several aspects are important when drawing class diagrams: The generalizations induce hierarchical components (*hierarchies*) of the graph, thus each hierarchy must be drawn such that all arcs run in the same direction; there are no restrictions on how the edges in the set E must be drawn. We call a drawing of G that satisfies these requirements a *mixed-upward* drawing. Moreover, the number of crossings between relationships should be small, generalizations belonging to different inheritance hierarchies should never cross, one hierarchy should not enclose another hierarchy, and the area covered by the drawing should be small.

For a clear visualization of the specific combination of hierarchical and non-hierarchical components in UML class diagrams, we put special emphasis on meeting a balanced mixture of the following aesthetic criteria:

- *Crossing minimization*
- *Bend minimization*
- *Orthogonal layout*
- *Uniform direction within each class hierarchy*: Arcs of a class hierarchy should point in a consistent direction.
- *No nesting of one class hierarchy within another*: A class hierarchy is not enclosed by a circle (in the undirected sense) of arcs of a different hierarchy.