

1 tikz-opm: creating Object Process Methodology diagrams with Tikz

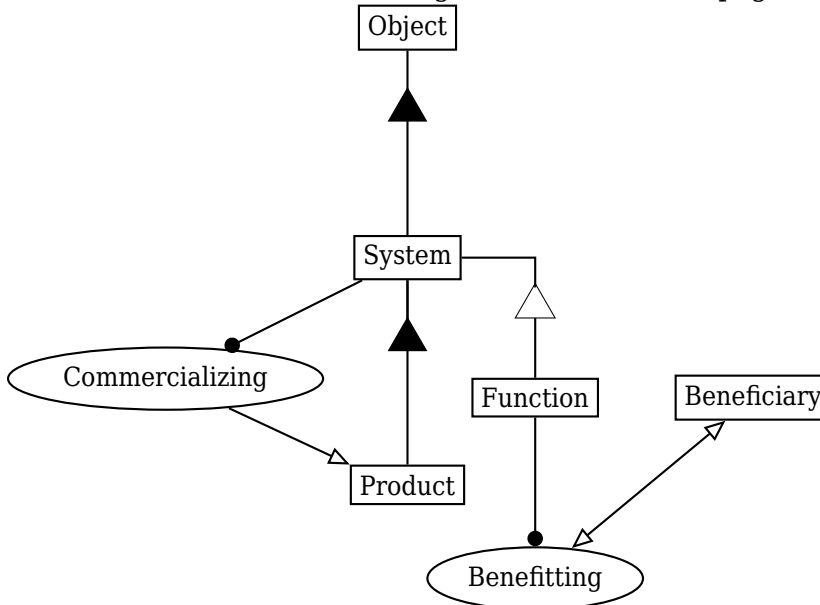
The tikz-opm package allows you to create nice OPM diagrams using a number of relatively simple commands.

The package does not attempt to do any clever things - I am not clever enough to encode smart placement of things like the aggregation symbol below the object it applies to. Sorry.

But it does allow you to most of the important things from OPM displayed in your diagram, so that is pretty good.

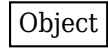
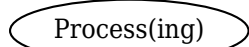
First we start out with the picture from the front of the OPM book:

Listing 1: OPM book front page

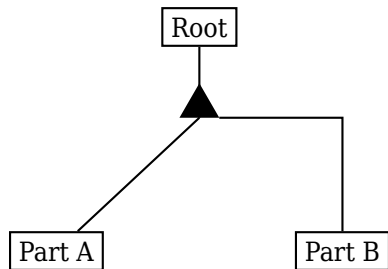


```
\begin{tikzpicture}
  \node [opmobject] (object) {Object};
  \node [opmaggregation, below=of object] (object-aggr) {};
  \path (object) edge (object-aggr);
  \node [opmobject, below=of object-aggr] (system) {System};
  \path (object-aggr) edge (system);
  \node [opmprocess, below left=of system] (commercializing)
  {Commercializing};
  \path (system) edge [opmagent] (commercializing);
  \node [opmaggregation, below=of system] (system-aggr) {};
  \node [opmobject, below=of system-aggr] (product) {Product};
  \path (system) edge (system-aggr) edge (product);
  \path (commercializing) edge[opmoutput] (product.north west);
  \node [opmgeneralization, below right=25pt of system, xshift=10pt] (system-gen) {};
  \node [opmobject, below=15pt of system-gen] (function) {Function};
  \draw [thick] (system.east) -| (system-gen.north);
  \path (system-gen.south) edge (function);
  \node [opmprocess, below=of function, yshift=-20pt] (benefitting) {Benefitting};
  \path (function) edge[opmagent] (benefitting);
  \node [opmobject, right=of function] (beneficiary) {Beneficiary};
  \path (benefitting) edge[opmeffect] (beneficiary);
\end{tikzpicture}
```

2 Catalog of OPM symbols

	<code>\node [opmobject] (object) {Object};</code>
	<code>\node [opmprocess] (processing) {Process(ing)};</code>

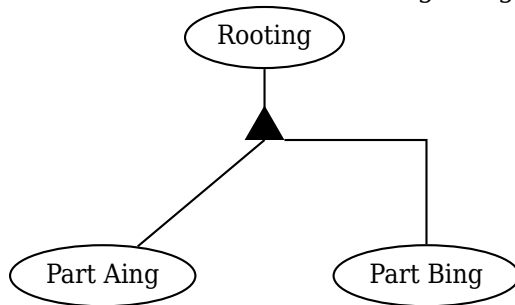
Listing 2: Aggregation for object



```
\node [opmobject] (root) {Root};
\node [opmaggregation, below=of root] (root-aggr) {};
\path (root) edge (root-aggr);
\node [opmobject, below left=of root-aggr] (part-a) {Part A};
\node [opmobject, below right=of root-aggr] (part-b) {Part B};
\path (root-aggr) edge (part-a);
\draw [thick] (root-aggr.east) -| (part-b.north);
```

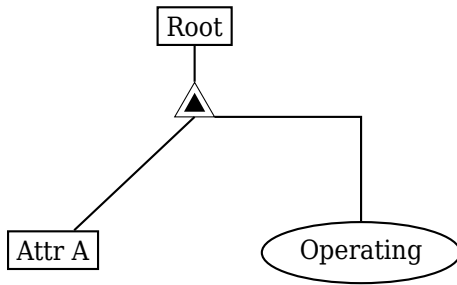
Note that in order to get the more traditional vertical/horizontal lines with right angles one has to use draw and the -| operator.

Listing 3: Aggregation for process



```
\node [opmprocess] (root) {Rooting};
\node [opmaggregation, below=of root] (root-aggr) {};
\path (root) edge (root-aggr);
\node [opmprocess, below left=of root-aggr] (part-a) {Part Aing};
\node [opmprocess, below right=of root-aggr] (part-b) {Part Bing};
\path (root-aggr) edge (part-a);
\draw [thick] (root-aggr.east) -| (part-b.north);
```

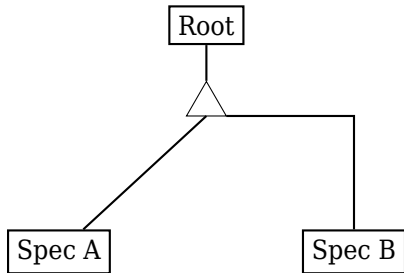
Listing 4: Exhibition



```
\node [opmobject] (root) {Root};
\node [opmexhibition, below=of root] (root-aggr) {};
\path (root) edge (root-aggr.north);
\node [opmobject, below left=of root-aggr] (part-a) {Attr A};
\node [opmprocess, below right=of root-aggr] (part-b) {Operating};
\path (root-aggr) edge (part-a);
\draw [thick] (root-aggr.east) -| (part-b.north);
```

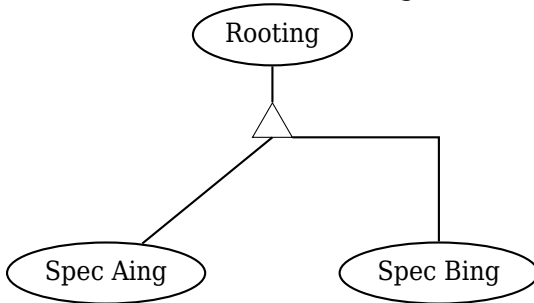
Note that one could equally well have a Rooting process instead of Root in exhibition.

Listing 5: Generalization for object



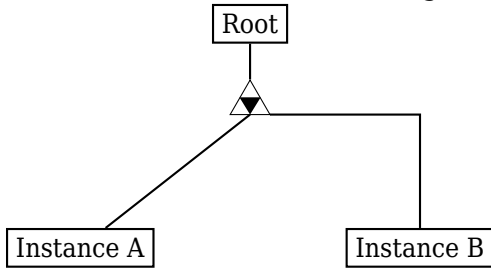
```
\node [opmobject] (root) {Root};
\node [opmgeneralization, below=of root] (root-aggr) {};
\path (root) edge (root-aggr.north);
\node [opmobject, below left=of root-aggr] (part-a) {Spec A};
\node [opmobject, below right=of root-aggr] (part-b) {Spec B};
\path (root-aggr) edge (part-a);
\draw [thick] (root-aggr.east) -| (part-b.north);
```

Listing 6: Generalization for process



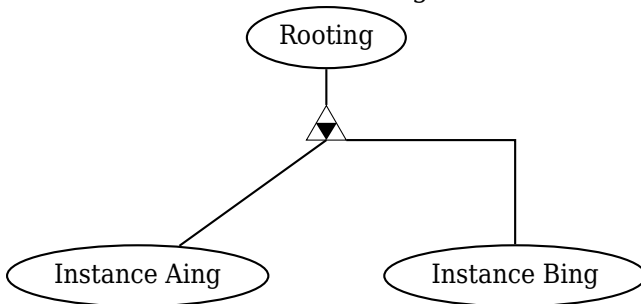
```
\node [opmprocess] (root) {Rooting};
\node [opmgeneralization, below=of root] (root-aggr) {};
\path (root) edge (root-aggr.north);
\node [opmprocess, below left=of root-aggr] (part-a) {Spec Aing};
\node [opmprocess, below right=of root-aggr] (part-b) {Spec Bing};
\path (root-aggr) edge (part-a);
\draw [thick] (root-aggr.east) -| (part-b.north);
```

Listing 7: Instantiation for object



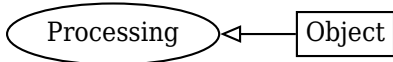
```
\node [opmobject] (root) {Root};
\node [opminstantiation, below=of root] (root-aggr) {};
\path (root) edge (root-aggr.north);
\node [opmobject, below left=of root-aggr] (part-a) {Instance A};
\node [opmobject, below right=of root-aggr] (part-b) {Instance B};
\path (root-aggr) edge (part-a);
\draw [thick] (root-aggr.east) -| (part-b.north);
```

Listing 8: Instantiation for process



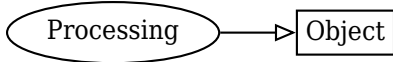
```
\node [opmprocess] (root) {Rooting};
\node [opminstantiation, below=of root] (root-aggr) {};
\path (root) edge (root-aggr.north);
\node [opmprocess, below left=of root-aggr] (part-a) {Instance Aing};
\node [opmprocess, below right=of root-aggr] (part-b) {Instance Bing};
\path (root-aggr) edge (part-a);
\draw [thick] (root-aggr.east) -| (part-b.north);
```

Listing 9: Consumption



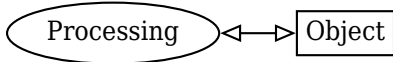
```
\node [opmprocess] (processing) {Processing};
\node [opmobject, right=of processing] (object) {Object};
\path (processing) edge[opmconsumes] (object);
```

Listing 10: Result



```
\node [opmprocess] (processing) {Processing};
\node [opmobject, right=of processing] (object) {Object};
\path (processing) edge[opmyields] (object);
```

Listing 11: Effect



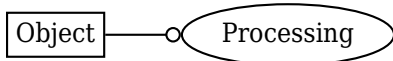
```
\node [opmprocess] (processing) {Processing};
\node [opmobject, right=of processing] (object) {Object};
\path (processing) edge[opmaffects] (object);
```

Listing 12: Agent



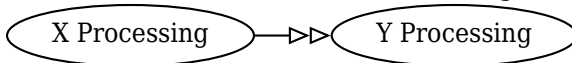
```
\node [opmprocess] (processing) {Processing};
\node [opmobject, left=of processing] (object) {Object};
\path (object) edge[opmhandles] (processing);
```

Listing 13: Instrument



```
\node [opmprocess] (processing) {Processing};
\node [opmobject, left=of processing] (object) {Object};
\path (processing) edge[opmrequires] (object);
```

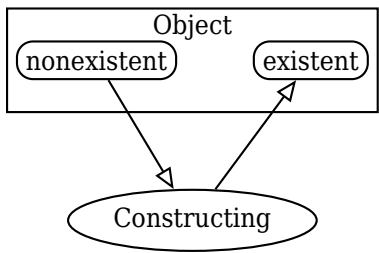
Listing 14: Invocation



```
\node [opmprocess] (x-processing) {X Processing};
\node [opmprocess, right=of x-processing] (y-processing) {Y Processing};
\path (x-processing) edge[opminvokes] (y-processing);
```

Note that the version with two arrow heads have been chosen since it is easier to work with than with the lightning zig-zag line that OPM uses in the book.

Listing 15: State



```

\node [opmstate] (nonexistent) {nonexistent};
\node [opmstate, right=of nonexistent] (existent) {existent};
\node [opmobject, fit=(nonexistent) (existent), text depth=5ex] (object) {Object};

\node [opmprocess, below=of object] (constructing) {Constructing};
\path (constructing) edge[opmconsumes] (nonexistent);
\path (constructing) edge[opmyields] (existent);

```

This is not optimal, so I could do with some help on making this drawing nicer. Using `text depth=5ex` is a very crude way of scaling the object box. Putting the `text depth` inside a Tikz style does not help *unless* one ensures that it is called after the `fit` key.