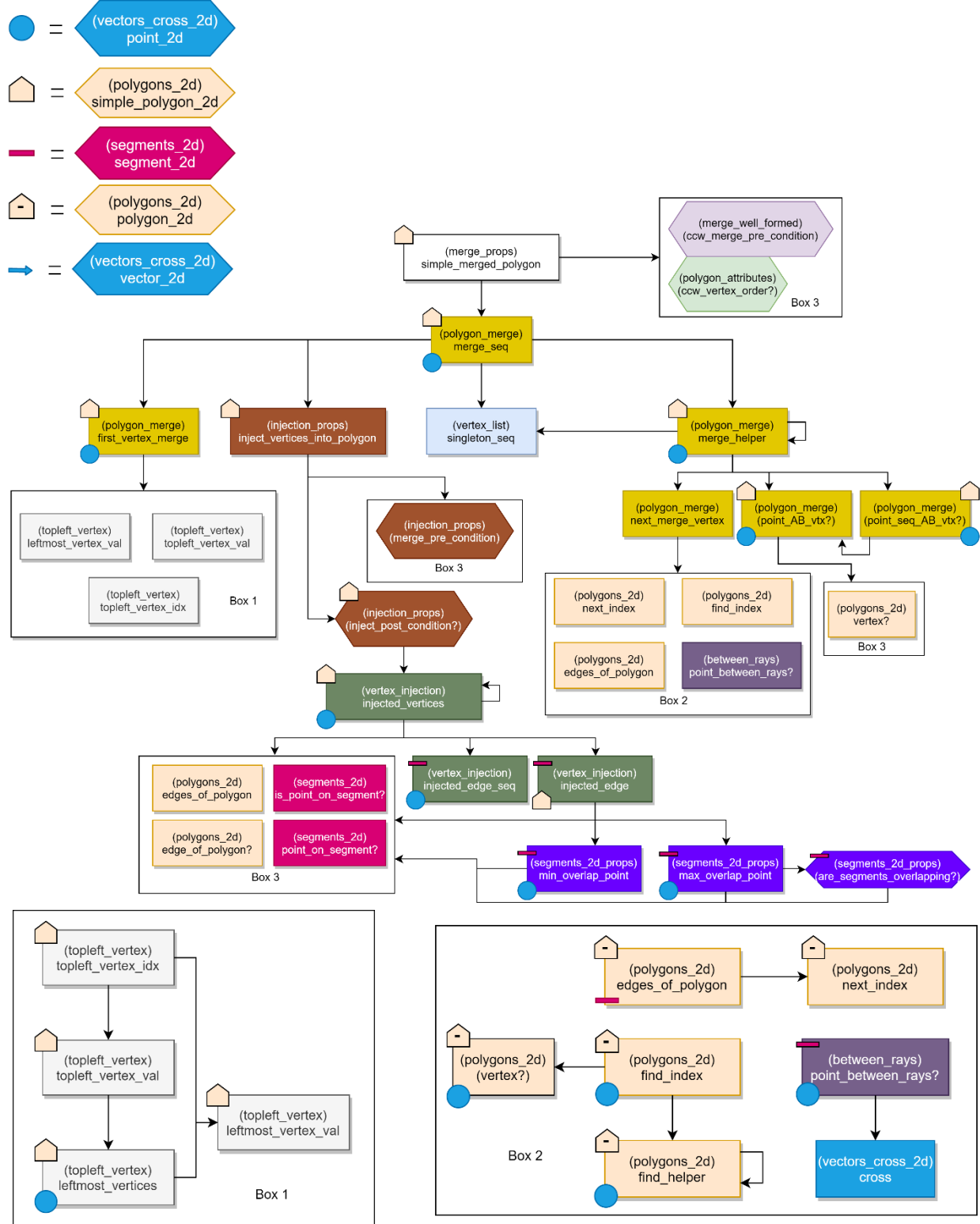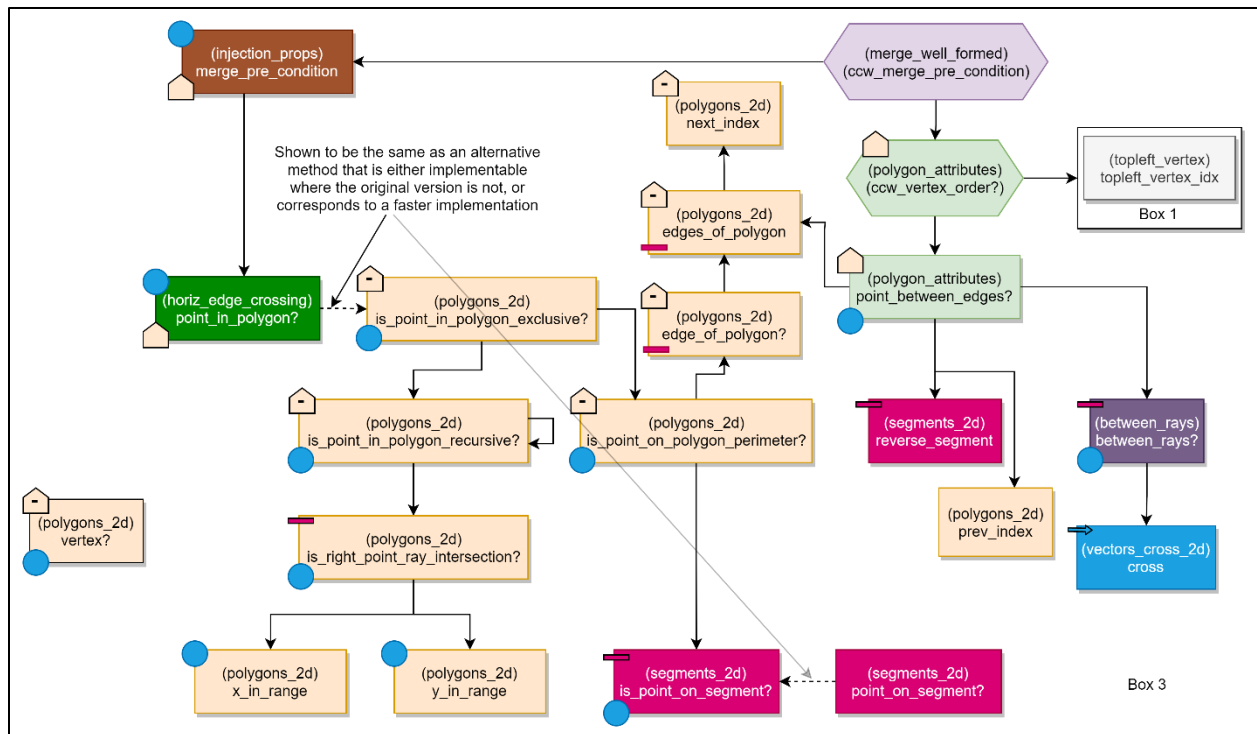The central function 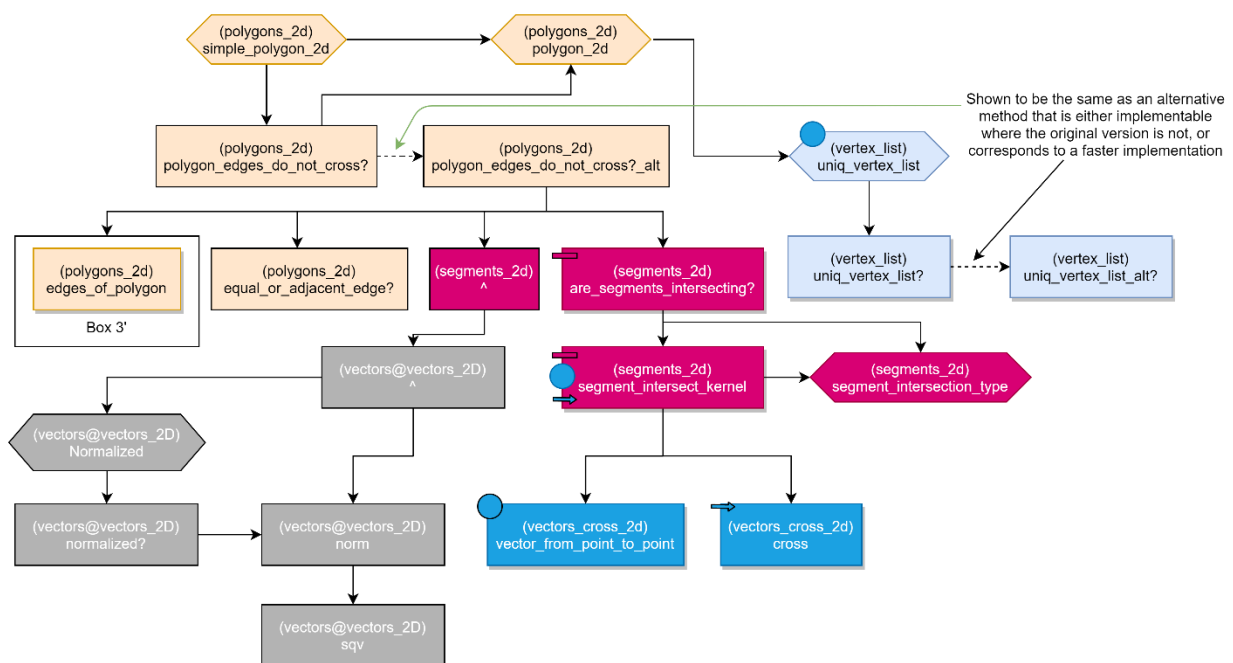of the `polygon_merge` library is `simple_merged_polygon` from the `merge_props` theory. The following figures show the hierarchy of dependencies of that function, ignoring only overloaded operators. This information is intended to be especially useful for implementations of the specification for `simple_merged_polygon`. The legend at top shows shortcut notation for the most common dependencies.

(injection_props) merge_pre_condition

(merge_well_formed) (ccw_merge_pre_condition)

(polygons_2d) next_index

(polygon_attributes) (ccw_vertex_order?)

(topleft_vertex) topleft_vertex_idx
Box 1

Shown to be the same as an alternative method that is either implementable where the original version is not, or corresponds to a faster implementation

(polygons_2d) edges_of_polygon

(polygon_attributes) point_between_edges?

(horiz_edge_crossing) point_in_polygon?

(polygons_2d) is_point_in_polygon_exclusive?

(polygons_2d) edge_of_polygon?

(segments_2d) reverse_segment

(between_rays) between_rays?

(polygons_2d) is_point_in_polygon_recursive?

(polygons_2d) is_point_on_polygon_perimeter?

(polygons_2d) prev_index

(vectors_cross_2d) cross

(polygons_2d) vertex?

(polygons_2d) is_right_point_ray_intersection?

(polygons_2d) x_in_range

(polygons_2d) y_in_range

(segments_2d) is_point_on_segment?

(segments_2d) point_on_segment?

Box 3

Dependencies for the **simple_polygon_2d** type:

(polygons_2d) simple_polygon_2d

(polygons_2d) polygon_2d

(polygons_2d) polygon_edges_do_not_cross?

(polygons_2d) polygon_edges_do_not_cross?_alt

(vertex_list) uniq_vertex_list

Shown to be the same as an alternative method that is either implementable where the original version is not, or corresponds to a faster implementation

(vertex_list) uniq_vertex_list?

(vertex_list) uniq_vertex_list_alt?

(polygons_2d) edges_of_polygon
Box 3'

(polygons_2d) equal_or_adjacent_edge?

(segments_2d) ^

(segments_2d) are_segments_intersecting?

(vectors@vectors_2D) ^

(segments_2d) segment_intersect_kernel

(segments_2d) segment_intersection_type

(vectors@vectors_2D) Normalized

(vectors@vectors_2D) normalized?

(vectors@vectors_2D) norm

(vectors@vectors_2D) sqv

(vectors_cross_2d) vector_from_point_to_point

(vectors_cross_2d) cross

Dependencies for the **segment_2d**, **vector_2d**, and **point_2d** types:


```
segment_2d: TYPE =
    [#
      p1: point_2d,
      p2: {p: point_2d | p /= p1}
    #];
```

```
vector_2d: TYPE = Vect2;
point_2d: TYPE = Vect2;
```